University of Nottingham
UK | CHINA | MALAYSIA

Machine Learning and Metaheuristic Methods for Optimizing Dynamic Truck Dispatching in Container Ports

Thesis submitted to the University of Nottingham for the degree of
Doctor of Philosophy, November 2023.

Xinan Chen

20199226

Supervised by

Ruibin Bai
Rong Qu

Signature Xinan Chen.

Date 19 / 2 / 2024

Abstract

Container port freight constitutes a pivotal component of contemporary global logistics, playing a decisive role in the convoluted matrix of international trade and supply chain management. Given the profound impact of container port operations on global trade dynamics, optimizing the transfer efficiency of container trucks within ports is of paramount significance. The efficiency of truck dispatching stands out as one of the core determinants determining port operational efficiency, thereby necessitating novel and advanced approaches to tackle the prevalent constraints and bottlenecks.

This paper explores machine learning (ML) methodologies to effectively navigate the complexities and uncertainties inherent in container port dynamic truck dispatching. Traditional methods in this domain frequently encounter limitations, particularly in their inability to adapt to real-time changes and handle the uncertainties characteristic of port operations. These conventional approaches, often reliant on static parameters, struggle to reflect port environments' constantly evolving and unpredictable nature accurately. In addressing these challenges, ML stands out as a highly suitable alternative, owing to its capacity to learn from vast datasets, adapt to novel scenarios, and make informed decisions under uncertain conditions.

The PhD project presented here is centered on two primary applications of ML: firstly, the development of innovative techniques for the generation and optimization of truck dispatching strategies, and secondly, the enhancement of operational efficiency and responsiveness amidst the fluctuating dynamics of port activities. By harnessing the adaptive and predictive capabilities of ML, this study aims to forge a more dynamic, responsive, and intelligent dispatching framework, one that is adept at overcoming the multifaceted uncertainties and fluidity inherent in contemporary container port operations.

Another facet of this PhD project seeks to augment the precision of evaluating truck dispatching strategies through refined ML methods. This segment is grounded in the aspiration to render the evaluation of diverse dispatching strategies more coherent and accurate, allowing for a meticulous assessment of their real-world efficacy and impact. The advancement in evaluation methodologies is anticipated to provide a nuanced understanding of individual dispatching strategies' intrinsic merits and demerits, contributing to developing more robust, effective, and tailor-fitted solutions.

Beyond these research works, this thesis delves into a comprehensive examination of the inherent challenges and opportunities residing within the interface of machine learning and truck dispatching. It elucidates the prospective advancements and innovations that machine learning can bring forth in optimizing dispatching mechanisms and strategies, emphasizing its potential to revolutionize container port logistics. This PhD research strives to discover new pathways for elevating operational efficiency and strategic intelligence in container port logistics through systematically synthesizing machine learning insights and domain-specific expertise.

The implications of this study are manifold, projecting substantial contributions to the enhancement of container port operational paradigms and global logistics frameworks. By incorporating machine learning into dynamic truck dispatching, this thesis aims to enhance efficiency and adaptability in the field, offering a more strategic approach to tackling the complexities and uncertainty involved. The findings and insights from this research are poised to provide valuable perspectives and pragmatic solutions for practitioners, policymakers, and academics, converging towards a more resilient, agile, and sustainable future in container port logistics and beyond.

Acknowledgements

First and foremost, I express my deepest gratitude to my esteemed supervisors, Prof. Ruibin Bai and Prof. Rong Qu. Their consistent patience, invaluable feedback, and encouragement have been instrumental throughout this journey. I vividly recall our first online interaction during the interview; their kindness was palpable and reassuring. I extend my deepest gratitude to Prof. Bai for his timely guidance regarding the PhD scholarship application process. Without his encouragement, I might not have even considered pursuing a PhD degree. I am very fortunate to have encountered such an excellent mentor as Prof. Bai during my PhD journey. I also hope Prof. Bai's research continues to flourish and achieve greater success! Prof. Qu, on the other hand, has been an unwavering source of inspiration in my academic journey. I am profoundly thankful to her for allowing me to delve into the world of academic reviewing. Her recent achievement of being promoted to full professor is a testament to her dedication and prowess. Congratulations, Prof. Qu!

A special place in my heart is reserved for my parents, Bin Chen and Li Li. Their unyielding faith in me, despite the extended academic path I chose, has been the cornerstone of my perseverance. Their support has been nothing short of a boon, especially considering the decade I dedicated to academia without pursuing a conventional career.

Extending the circle of gratitude, I would like to acknowledge my in-laws, Yuezhang Dong and Xiajun Chen. Their steadfast support and guidance, both academically and in life's nuanced lessons, have been invaluable. Their wisdom, shared often through their lived experiences and exemplary actions, has illuminated many paths for us. It is due to their unwavering dedication and support that my wife and I currently find ourselves cherishing such joyous moments in life.

I extend my warm appreciation to Prof. Dongfang Liang and Dr. Wenhua

Zhao. Their profound insights and advice have shaped my academic trajectory and provided guidance for life beyond academia. I fondly remember the delightful formal gatherings in Cambridge, which they graciously invited me to.

I am deeply indebted to my friends Dr. Lei Li, Dr. Ling Jiang, and Dr. Xiaoping Jiang. As the first friends I made during my PhD journey, they have been my pillars of strength, consistently encouraging me to push forward. I wholeheartedly wish them continued success and brilliance in their respective endeavors.

Lastly, but most significantly, my heartfelt gratitude goes to my wife, Jing Dong. The odyssey of pursuing a PhD is inherently challenging and solitary. However, with her by my side, every hurdle seemed surmountable, every challenge an adventure. Her unwavering support and companionship have transformed this journey into a cherished chapter of my life. My aspiration is to remain by her side, cherishing, supporting, and loving her every step of the way.

This thesis is wholeheartedly dedicated to my beloved wife, Jing Dong.

Contents

List of Tables

List of Figures

Abbreviations

AGP Arithmetic Genetic Programming.

AGV Automated Guided Vehicle.

CD-GPHH Cooperative Double-Layer Genetic Programming Hyper-heuristic.

DNN Deep Neural Network.

DRL-EHH Deep Reinforcement Learning-assisted Ensemble Hyper-heuristics.

DRL-GPEHH Deep Reinforcement Learning-assisted Genetic Programming Ensemble Hyper-heuristics.

DRL-GPHH Deep Reinforcement Learning-assisted Genetic Programming Hyper-heuristic.

DRL-HH Deep Reinforcement Learning-based Hyper-heuristics.

DT Decision Tree.

FETT Fixed Extra Travel Time.

FIPT Fixed Intersection Passing Time.

GP Genetic Programming.

GPRL-H Genetic Programming and Reinforcement Learning Hybrid.

GPS Global Positioning System.

IoT Internet of Things.

LGP Logic Genetic Programming.

NN-GP Neural Network Assisted Genetic Programming.

QC Quay Crane.

QCW Quay Crane Waiting Time.

RL Reinforcement Learning.

TEU Twenty-foot Equivalent Unit.

TPH TEUs Processed per Hour.

TSGP Transformer-Surrogate Genetic Programming.

YC Yard Crane.

Chapter 1

Introduction

A container port, also known as a container terminal, is a dedicated facility where cargo containers are transshipped between various modes of transport, such as container ships, trucks, and rail, for onward transportation. Equipped with specialized heavy machinery, including cranes and forklifts, these ports are designed to handle shipping containers efficiently. They play a pivotal role in international trade and the global supply chain, serving as key hubs in the movement of goods worldwide. The efficient operation and management of container ports are crucial to ensuring an uninterrupted cargo flow, significantly impacting economic activities and global trade.

In this context, container ports emerge as critical nodes within the global logistics and supply chain network, acting as pivotal junctures where goods transition between maritime vessels and land-based transportation. At the heart of this intricate network lies dynamic truck dispatching, a key operational component that dictates the efficiency and fluidity of freight movement within and beyond the port's boundaries. The optimization of truck dispatching is driven by the need to alleviate operational bottlenecks and enhance port throughput, contributing significantly to the robustness and resilience of international trade mechanisms. This need becomes increasingly critical as trade volumes escalate, highlighting the complexities and challenges in logistical operations.

In the unfolding sections of this introduction, a comprehensive exposition is provided to ground readers in the multifaceted aspects of this research. **Section 1.1: Background and Motivation** establishes the overarching context and elucidates the motivations underpinning this study, emphasizing the integral role of container ports in the global trade ecosystem and the operational imperatives of dynamic truck dispatching. Following this, **Section 1.2: Problem Distribution and Formulation** offers a meticulous exploration of the inherent challenges and nuances associated with truck dispatching in container ports, thereby articulating the research problem precisely. **Section 1.3: Research Objectives and Scope** delineates the specific objectives and the analytical boundaries guiding this inquiry, clarifying the research's investigative trajectory and methodological alignment. Progressing to **Section 1.4: Significance of the Study**, the potential impact and contributions of the research findings are underscored, projecting their relevance and applicability in advancing operational paradigms within container port logistics. Finally, **Section 1.5: Thesis Structure** presents an organized overview of the ensuing chapters and their thematic focus, preparing readers for a structured and in-depth engagement with machine-learning-based approaches to optimizing dynamic truck dispatching at container ports.

1.1 Background and Motivation

Container ports, the nexus between maritime and land-based transportation, hold significant importance in the global trade and logistics ecosystem (Hall, 2009). These pivotal converging points are indispensable components in international trade, playing a vital role in the transportation and transshipment of goods across continents. The increasing globalization and intensification of trade activities accentuate the significance of container ports, necessitating innovative solutions to accommodate the burgeoning demands and to address the inherent complexities in logistical operations (Berkes et al., 2006).

The orchestration of operations within container ports involves many variables and components, among which dynamic truck dispatching stands prominently as a determinant of overall operational efficiency (Chen et al., 2019). It encompasses the real-time allocation and scheduling of trucks for container transfers within the port premises, which is integral to preventing congestion, minimizing dwell times, and ensuring seamless container movements. The optimization of these dispatching processes is paramount to achieving operational agility and enhancing the port's throughput capabilities (Talley, 2006b).

However, the multifaceted nature of port operations renders traditional truck dispatching methodologies based on human operators increasingly inadequate, as they often struggle to reconcile the dynamic and variable elements inherent in port logistics. This inadequacy amplifies the operational bottlenecks and hampers the adaptability and responsiveness of dispatching strategies to evolving demands and operational contingencies. Thus, there is a pressing need for advanced and adaptive solutions that can adeptly navigate the intricate landscape of container port operations and offer optimized dispatching strategies.

Given the above challenges, machine learning emerges as a potential catalyst for innovation in dynamic truck dispatching. The capabilities of machine learning extend beyond mere predictive analytics; they encompass learning and adapting to changing environments, offering nuanced and intelligent solutions to complex problems (Chen et al., 2016). The incorporation of machine learning models in truck dispatching can usher in a new era of operational intelligence and efficiency, enabling the development of dispatching strategies that are attuned to real-time operational variables and are capable of making informed and optimized decisions.

This research is anchored in the aspiration to explore and elucidate the transformative potential of machine learning in redefining dynamic truck dispatching within container ports. It is motivated by the prospect of unearthing insights and developing methodologies that could significantly augment the operational performance and strategic adaptability of container ports. By delving into the confluence of container port logistics and machine learning, this study aims to contribute to the evolving discourse in this domain and to proffer solutions that are both innovative and pragmatic, fostering operational resilience and excellence in container port logistics.

Furthermore, the broader implications of optimizing dynamic truck dispatching reverberate beyond the confines of container ports. Enhanced dispatching strategies have the potential to alleviate the logistical strains on the entire supply chain, contributing to the sustainability and robustness of global trade networks. Therefore, this study is not just an academic endeavor; it is a pursuit to drive meaningful advancements in global logistics and to catalyze positive transformations in international trade dynamics.

1.2 Problem Distribution, Formulation and Simulation

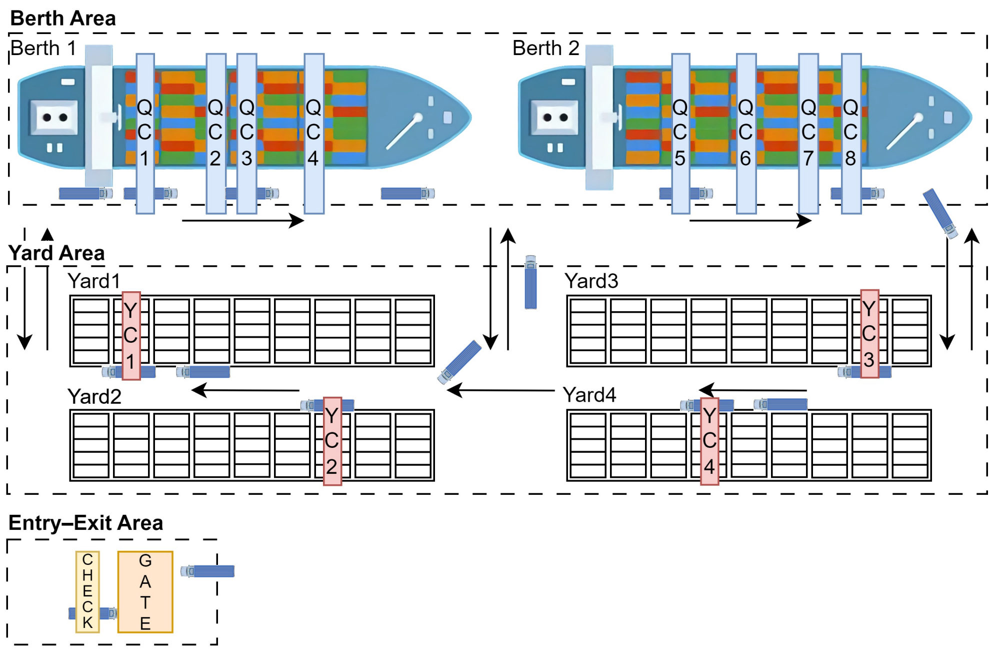


Figure 1.1: Sample Map of a Typical Container Port

1.2.1 Problem Description

Container ports are pivotal nexuses in the vast labyrinth of global trade networks, serving as the operational epicenters that orchestrate the intricate ballet of international commerce. A closer inspection of a typical container port, delineated in Fig. 1.1, unveils a sophisticated tableau of multifarious processes (Li et al., 2012). This intricacy interweaves the strategic areas of berths and yards, with container trucks emerging as the indispensable conduits connecting these dynamic realms and sustaining the relentless rhythm of logistical activities.

Within this complex logistical panorama, a marine container terminal bifurcates into three cardinal segments: the berth area, yard area, and the entry–exit area, with the illustrative example encompassing five berths. Along these berths, quay cranes (QCs) are meticulously stationed, dedicated to loading and unloading containers onto and from the waiting vessels. These QCs maneuver along a single rail designed parallel to the berth line, enabling fluid transitions between different berths, albeit with a constraint that prevents overlapping movements.

Delving deeper, the yard area emerges as a transient sanctuary for container storage, organized into uniformly sized yard blocks, each distinguished by a unique ID such as A1, or C4, situated at the heart of the yard area. These blocks are typically overseen by one or two-yard cranes (YCs) or analogous equipment, instrumental in executing the loading and unloading tasks, with queues inherently forming due to the operational limitation of handling only unit-sized containers at any given time.

Strategically, berths are anchored in deep-sea water areas, linked to the yards by a meticulously designed road network, bridging the shallow water regions and punctuated by road segments and intersections. This network is the arterial route for trucks, specifically inner trucks, transporting containers between QCs and YCs, adhering to stringent traffic regulations, ensuring operational safety, and mitigating congestion.

Table 1.1: Example of Work Instructions

ID	**ContainerID**	**Src**	**Dst**	**Type**	**TEUs**	**Ton**	**TwinID**
1731	FCIU3705890	CR12	J4	DSCH	2	17	0
1287	ECMU9249162	Q2	CR1	LOAD	1	23	1514
137	NYKU2797417	Q5	CR7	LOAD	2	15	0
1514	TCLU5546292	CR12	CR15	DSCH	1	19	1287

The environment surrounding these ports is a dynamic nexus of activity, demanding precise coordination of tasks related to loading and unloading containers. This complex process intertwines ships, berths, and yards.

Considering the varied container sizes and specific task requirements—as detailed in Table 1.1—the development of accurate, responsive, and adaptable truck dispatching algorithms becomes a critical operational need.

In Table 1.1, 'ID' is the task identifier, and 'ContainerID' is each container's unique identifier. 'Src' and 'Dst' indicate the container's source and destination locations. The tasks fall into two categories: 'DSCH' for unloading from ship to yard and 'LOAD' for loading from yard to ship. 'TEUs' represents the container's size, while 'Ton' measures its weight. Since containers are typically 40-foot or 20-foot, each truck can carry either two small or one large container. To accommodate this, small containers are paired to create a combined 'bind task', identified by 'TwinID'.

These tasks vary in weight, start and end locations, types of operations, container sizes, and considerations for bundled containers. The dispatching algorithms, therefore, are vital in predicting and mitigating logistical challenges and operational bottlenecks. They play an essential role in streamlining the transition of goods and improving overall operational efficiency, thus forming a crucial element in the complex fabric of port operations.

The diverse nature of operational elements within container ports, including disparate capacities of operational cranes, a spectrum of regulatory frameworks, and a continuous quest for enhanced precision and real-time responsiveness, are critical. These elements accentuate the operational challenges and underscore the criticality of developing sophisticated, contextually intelligent dispatching algorithms capable of navigating the diverse and dynamic terrains of container ports with enhanced efficacy.

In the contemporary operational landscape fraught with complexities and inherent dynamism, the evolution toward advanced surrogate simulators is pivotal. These simulators are instrumental in mimicking the nuanced operational processes of ports and act as fertile grounds for the development, refinement, and exhaustive evaluation of innovative dispatching strategies, deciphering their impact on enhancing operational throughput and efficiency.

Combining machine learning models within these simulative environments opens up unprecedented avenues for operational enhancement. It promises a transformative shift towards higher simulation accuracy, optimized evaluative mechanisms, and the cultivation of profound, data-driven insights, enabling the synthesis of robust, resilient, and precision-oriented dispatching strategies. The intrinsic capabilities of machine learning in discerning intricate patterns, formulating predictive models, and facilitating adaptive learning stand as beacons of potential revolutionary advancements in container port operational paradigms.

This thesis traverses the intricate interplay between machine learning and the dynamic ecosystems of container ports. It seeks to explore and elucidate the transformative potential of machine learning in optimizing truck dispatching algorithms and enhancing simulation environments, aiming to inject new vigor into container port operations. It strives to unearth innovative solutions, methodologies, and insights that can redefine operational strategies, catapulting container port logistics to new heights of operational excellence, resilience, and adaptability.

In such a complex landscape, the algorithms play a pivotal role by optimally allocating trucks to QCs and YCs, aiming to minimize the idleness and waiting times integral to the container handling processes, thus optimizing the overall ship turnaround efficiency. The operational capacities of QCs, capable of handling two small or one large container, and Yard Crane (YC)s, limited to one container per operation, accentuate the complexity of the operational tapestry, warranting precise, adaptive solutions.

Moreover, the essence of truck dispatching within container ports cannot be understated. It is the cornerstone of logistic operations, influencing the timely and efficient transfer of containers between ships and yards, impacting the turnover time of ships, and by extension, the overall throughput of the port. A delay in truck dispatching can cascade into substantial operational delays, elevating the dwell time of containers and ships, leading to increased operational costs and reduced port competitiveness. Thus, refining truck dispatching strategies through machine learning not only stands as a technological advancement but also as a crucial element in enhancing the economic viability and competitive edge of container ports in the global landscape.

This exploration seeks to elucidate the pivotal role of machine learning in augmenting decision-making processes, advancing predictive accuracy, and bolstering the adaptive capacities of dispatching algorithms within container port operations. The research delves into the operational complexities, identifies underlying inefficiencies, and aims to design innovative and sustainable solutions that embody a new paradigm of robustness, agility, and efficiency for container ports. Through a deepened engagement with machine learning technologies, this thesis is committed to transforming container port truck dispatching into a more efficient, intelligent, and robust process. These enhancements are imperative to fulfill the exacting requirements of container port companies that strive for unwavering operational efficiency amidst the challenges posed by varying environmental conditions and the intricacies of complex operational scenarios.

1.2.2 Problem Formulation

The challenge of effectively orchestrating port truck dispatching is one steeped in complexity and laden with uncertainties. The intricate dance of logistics involves multiple variables, such as the transitory timings of container trucks, the meticulous operations of QCs, and YCs, and the unpredictable influx of external container trucks—all coordinately converge to shape the overarching operational tapestry of the port. The interplay of these multifarious factors creates a dynamic environment, making pursuing a comprehensive mathematical model for this problem a daunting expedition, fraught with impediments to obtaining quality, precise solutions through traditional solving methodologies.

Acknowledging these complexities, this thesis endeavors to navigate the intricate labyrinth of the port truck dispatching problem and seeks to construct a simplified yet robust mathematical model. This model, by virtue of its inherent simplicity, strives to embody the quintessential attributes of the problem, providing a foundational framework to delve into the intricate undercurrents and elaborate nuances inherent in the dispatching dynamics. Through this foundational model, the objective is to elucidate the myriad aspects of the port truck dispatching scenario, thereby enhancing comprehension and enabling a more nuanced appreciation of the multifaceted nature of the problem. This approach serves as a precursor to further in-depth explorations and sophisticated analyses aimed at unearthing innovative solutions and optimizing strategies to address the complexities intrinsic to the port truck dispatching ecosystem.

The problem can be formally delineated as follows. An abstract container terminal is depicted as a directed graph, denoted by $G = (A, C)$, where $C = Q \cup Y$ constitutes the nodes representing the work operation points for all tasks. The sets Q and Y encompass all QCs and YCs, respectively. The set A consists of direct driving connections between distinct nodes. The truck depot, represented by d, is the point from which all trucks depart at the commencement of the operation and return upon completion of all tasks. The set $V = \{v_1, v_2, v_3, \ldots, v_m\}$ signifies the collection of m available trucks for allocation. A function $\tau(x, y)$ maps two disparate operation points, $x \in C$ and $y \in C$, to the time required to traverse from one point to the other, reflecting the actual terminal road network. The work instruction list encompasses all n transport tasks in $T = \{t_1, t_2, t_3, \ldots, t_n\}$. The container size for each task t_i is denoted by $size_i$. The source and destination nodes for a given t_i are represented by a_i and b_i, respectively, with $a_i, b_i \in C$. Based on the diverse types of source and destination nodes, ty_i is defined as the type of task i. $ty_i = 1$ signifies an unloading task, while $ty_i = 0$ corresponds to a loading task.

Within our problem framework, tasks are confined to transportation journeys exclusively between QCs and YCs. Consequently, a_i and b_i pertain to distinct crane-type node sets, either QCs or YCs. The maximum difference in task serial numbers, denoted as q, indicates the acceptable swapping order of unloading tasks (in this paper, $q = 3$, considering the practicalities). The start time of service for t_i at its source node is represented by s_i, while its completion time at the destination node is symbolized by e_i, where $s_i \in S = \{s_1, s_2, s_3, \ldots, s_n\}$ and $e_i \in E = \{e_1, e_2, e_3, \ldots, e_n\}$. Since a crane is required to either load or unload the container at the beginning and end of a task, the parameters d_i and h_i depict the operating time of t_i at the source and destination nodes, respectively, and their sum is r_i. The operation times at QCs and YCs are assumed to be stochastic and extracted from historical data.

To model the problem formally, the assignments of tasks to trucks are defined by the following binary variable in (1.1):

$$\alpha_{ij} = \begin{cases} 1 & t_j \text{ is assigned to } v_i \\ 0 & \text{otherwise} \end{cases} \tag{1.1}$$

The following auxiliary variable indicates whether t_k is serviced immediately after task t_j by truck v_i.

$$\beta_{ijk} = \begin{cases} 1 & t_k \text{ is served right after } t_j \text{ by } v_i \\ 0 & \text{otherwise} \end{cases} \tag{1.2}$$

The order of tasks belonging to a crane $c_i \in C$ is described by (1.3).

$$\gamma_{ijk} = \begin{cases} 1 & t_k \text{ is followed by } t_j \text{ in } c_i \\ 0 & \text{otherwise.} \end{cases} \tag{1.3}$$

The primary objective in truck dispatching problems for container terminals involves enhancing the company's profitability by increasing turnover and minimizing the waiting time of ships. To evaluate the extent to which this objective is accomplished, various metrics can be employed. In this study, we focus on the objective of *TEU per hour (TEU/h)*, which is a metric calculating the quantity of Twenty-foot Equivalent Units (Twenty-foot Equivalent Unit (TEU)s) processed hourly by all Quay Cranes (QCs) in use. The TEU, a standardized measure for containerized cargo, corresponds to a twenty-foot container's capacity. Port companies widely adopt this metric as a key indicator for benchmarking their operational efficiency against competitors. It is noteworthy that the TEU/h metric is analogous to the makespan employed in numerous scheduling problems when the task set remains constant. Consequently, our truck dispatching problem can be modeled as follows:

$$\max(\frac{\sum_{i=1}^{n} size_i}{\max(E) - \min(S)}) \tag{1.4}$$

$$\sum_{i=1}^{m} \alpha_{ij} = 1 \ \ \forall t_j \in T \tag{1.5}$$

$$\sum_{i=1}^{m} \sum_{k=1}^{n} \beta_{ijk} \leq 1 \forall t_j \in T \tag{1.6}$$

$$\sum_{j=l}^{n} \sum_{k=1}^{l} \gamma_{ijk} \leq q \cdot y_i \ \ l \in [1, n] \tag{1.7}$$

$$s_i = \max \begin{cases} \sum_{j=1}^{n} \sum_{k=1}^{m} \beta_{kji} \cdot (\tau(b_j, a_i) + e_j) \\ \tau(d, a_i) \cdot (1 - \sum_{j=1}^{n} \sum_{k=1}^{m} \beta_{kji}) \end{cases} \tag{1.8}$$

$$e_i = \max \begin{cases} \max(s_i, \sum_{j=1}^{n} \sum_{k=1}^{m} \gamma_{kji} \cdot e_j) + \tau(a_i, b_i) + r_i \\ \sum_{j=1}^{n} \sum_{k=1}^{m} \gamma_{kji} \cdot e_j + d_i \end{cases} \tag{1.9}$$

The objective delineated in (1.4) represents the average production rate per unit of time (hour), where $\max E$ and $\min S$ correspond to the completion time of the final task and the start time of the first task, respectively. The constraint articulated in (1.5) guarantees that each task is assigned exclusively to one truck. In contrast, the constraint in (1.6) ascertains that each task is succeeded by a maximum of one other task or none if it is the truck's final task. For each crane, constraint (1.7) following container terminal transportation rules, ensure that tasks involving the same crane cannot commence until the preceding task is concluded, except for the unloading tasks in QCs where the operational sequence can be interchanged between $sn = 3$ neighboring tasks. Constraints (1.8) and (1.9) calculate the start and end times of tasks, verifying that tasks initiate crane operation only after completing preceding task operations.

The task of adeptly managing truck dispatching within maritime container terminals holds the formidable distinction of being NP-hard, corroborated by its reducibility to the well-known vehicle routing problem (Chu et al., 2012). This inherently intricate classification implies that the quest for the optimal solution escalates into an exponentially computational endeavor as the dimensionalities of the problem magnify. Historical studies have predominantly leveraged metaheuristic approaches, predicated on the assumption that crane operation times r_i for tasks within the task set T exhibit a deterministic constancy. However, the operational landscape of container terminals is punctuated with inherent uncertainties and variances in crane operation times, rendering the prospect of absolute predictions a theoretical ideal rather than a pragmatic possibility.

Our investigative journey into utilizing mixed-integer programming (MIP) solvers, constructed on the bedrock of formulated problem constraints and parameters, encountered substantial impediments. The formidable NP-hard complexity intertwined with the stochastic problem's parameters converged to create a landscape fraught with computational challenges, thereby obfuscating the pathway to identifying feasible and optimal solutions. This intricate interplay between complexity and uncertainty necessitated a paradigmatic shift in our problem-solving approach.

Consequently, we have delineated the truck dispatching conundrum in container ports as an online optimization problem. Our exploration traverses the realms of heuristic-based methodologies, seeking to distill pragmatic and efficient solutions from the chaotic symphony of variables and constraints inherent in the dynamic environment of maritime container terminals. This heuristic lens endeavors to bridge the chasm of uncertainties and complexities, providing a navigable pathway to optimized dispatching solutions while embracing the intricate dance of variables within the operational tapestry of container ports.

Meanwhile, heuristic approaches are currently at the forefront of methodologies employed and promoted by most port companies. While heuristics may not guarantee the optimality of solutions, they bear a striking resemblance to the manual truck dispatching methods traditionally utilized by ports, allowing for a smooth transition to more advanced techniques. Moreover, due to the time-sensitive nature of port operations, where dispatching decisions must be rendered swiftly, heuristic methods align well with the immediacy required in intelligent port operations. Thus, this thesis prioritizes heuristic approaches, intending to progressively explore the application of machine learning in heuristic-based dynamic container port truck dispatching problems. The exploration will critically evaluate the effectiveness of these heuristics in operational scenarios, benchmark their performance against evolving machine learning paradigms, and propose novel strategies to enhance their efficiency and intelligence, aligning with the operational cadence and the complex decision-making milieu of contemporary ports.

1.2.3 Container Port Simulation

In the complex ecosystem of container port operations, simulators transcend basic algorithmic verification, providing an indispensable platform for thorough experimentation and analysis. Real-world ports are pulsating with a myriad of tasks that require precise timing and coordination for the seamless transfer of cargo. These operations are so intricate and tightly coupled that any disruption could lead to significant logistical setbacks and economic ramifications. Consequently, testing new algorithms directly in such a bustling environment is fraught with risks. Any unproven or suboptimal algorithm has the potential to cause delays, increase operational costs, or even jeopardize safety. This renders the real environment unsuitable as a primary testing ground for developmental algorithms.

Machine learning methods, inherently data-driven and adaptive, require a controlled yet complex environment where they can interact, learn, and evolve. The learning phase of these algorithms involves trial and error, necessitating a space where mistakes can be made without real-world consequences. Moreover, the iterative process of training machine learning models to achieve optimal performance requires large volumes of data and numerous simulation runs, which can only be feasibly conducted within a simulated environment.

Simulators provide a sandbox where the entire spectrum of container port operations can be recreated with a high degree of fidelity. They offer a virtual yet realistic representation of port operations, including vessel arrival patterns, container loading and unloading sequences, and the dynamic scheduling of trucks. In such a simulation, machine learning algorithms can be exposed to various scenarios, including peak loads, unpredictable delays, and scheduling conflicts, which are critical for training robust and resilient models.

Furthermore, simulators can accelerate time, allowing machine learning algorithms to experience years of operational data in days or hours, thus significantly speeding up the learning process. They can also mimic diverse operational conditions and exceptions, providing a comprehensive suite of tests that ensure the algorithm's reliability and adaptability to real-world variances.

In essence, simulators act as the crucible for refining the intelligence of machine learning models tailored for container port truck dispatching. They offer a risk-free environment to experiment, learn from errors, and systematically improve performance before any algorithm is deployed in the real world, thereby upholding the uninterrupted flow of commerce that container ports rely upon.

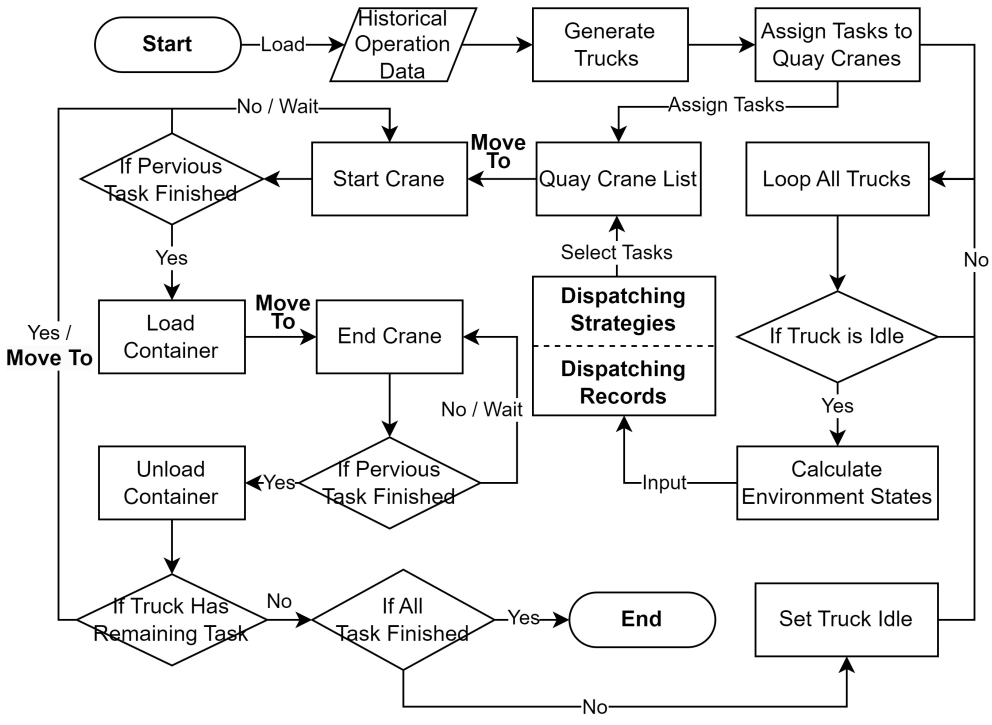


Figure 1.2: Event-based Port Simulator Flow Chart

Understanding the complexity of port operations, we aim to elucidate our approach to constructing a simulator that reflects the multifaceted operational dynamics inherent in container ports. As illustrated in Fig. 1.2, the developed simulator is structured to incorporate many operational components and events that are fundamental to the fluidity and functionality of port operations.

As shown in Fig. 1.3, simulators typically fall into two primary categories: time-stepped and event-based simulations. Time-stepped simulations are intricate, evaluating every operational entity at specific time intervals and forecasting subsequent events with high precision. This method, although extensive and detailed, demands substantial computational resources and time—around 200 times more than event-based simulations when the time span is set to one second, as observed in our experiments. Therefore, balancing precision and computational efficiency, an event-based simulation approach has been deemed optimal for our study, ensuring swift evaluations, training, and testing of algorithms.

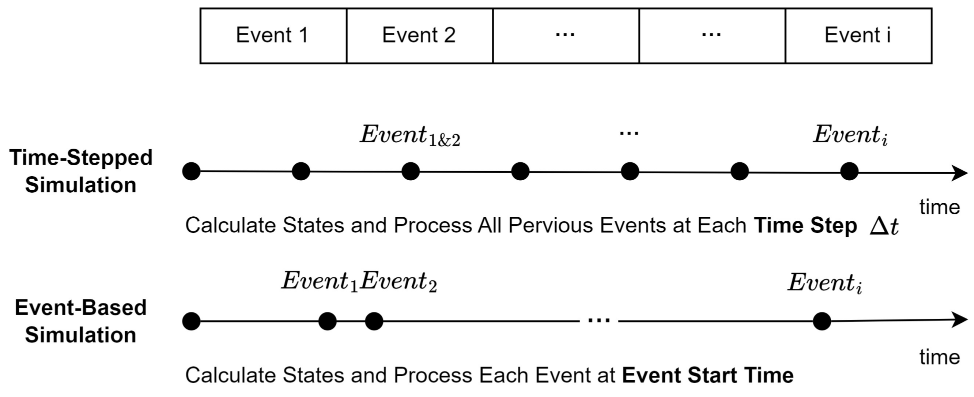

Event 1
Event 2
···
···
Event i
Time-Stepped Simulation
$Event_{1\&2}$
···
$Event_i$
time
Calculate States and Process All Pervious Events at Each **Time Step** Δt
Event-Based Simulation
$Event_1 Event_2$
···
$Event_i$
time
Calculate States and Process Each Event at **Event Start Time**

Figure 1.3: The Difference between Time-Stepped and Event-Based Simulation

Event-based simulations, a subtype of discrete-event simulations (DES), propel the simulation process by calculating and processing events selectively when critical incidents occur, thus offering an efficient method to simulate the multifarious nature of port operations. Our developed simulator operates on the principles of event-based simulation and replicates the operational intricacies of container port truck dispatching, focusing on principal events such as container loading, unloading, and intricate truck movements.

In the construction of our simulation model, initial considerations involve the incorporation of historical port operation data, allowing the simulator to assimilate an extensive range of variables such as task specifics, numerics of trucks, and their ensuing locations, thereby establishing an authentic initial state. This meticulous initiation process is fundamental as it ensures the simulation commences with a realistic representation of the port's operational environment.

Following this initialization, the simulation launches into its main iterative process, with a predominant focus on trucks that are in an idle state. This emphasis is integral to accurately representing operational dynamics, as the states of idle trucks are meticulously evaluated against environmental parameters. Subsequently, the calculated states are intricately interwoven with predefined dispatching strategies to ascertain appropriate tasks for each idle truck.

To undertake varying assessment tasks with optimum precision and relevance, our methodology strategically employs two distinct approaches, each meticulously tailored to align with the specific objectives of the assessment in question, for dispatching idle trucks within the simulation environment.

The initial approach is predominantly implemented when the primary objective is to scrutinize the efficiency and effectiveness of dispatching algorithms. In such scenarios, we strategically utilize the dispatching strategies, meticulously formulated and refined by the respective algorithms, as the blueprint to allocate tasks to idle vehicles. This approach is indispensable in enabling a profound exploration of the operational capabilities and potential limitations of the dispatching algorithms, facilitating an in-depth analysis of their operational proficiency and adaptability within the dynamic landscape of container port logistics.

In contrast, when our focus shifts towards evaluating the simulation's performance and accuracy comprehensively, our methodology adopts a more data-centric approach. Here, we meticulously reference the dispatching records encapsulated within the historical data. This approach ensures that the tasks delegated to the vehicles not only align with but also accurately mirror the historical trends and occurrences, providing a cohesive and authentic representation of operational realities. By adhering to historical precedents, this approach cultivates an environment of enhanced realism within the simulation, allowing for a more rigorous and contextual assessment of the simulation's fidelity and operational coherence.

Each of these approaches is meticulously calibrated to preserve the integrity and reliability of the assessment process, enabling a harmonious balance between theoretical formulations and practical implementations. They work in tandem to offer a multifaceted perspective on the interplay between dispatching strategies and operational dynamics, fostering a deeper understanding and facilitating the development of innovative solutions and strategies to navigate the complexities inherent in container port logistics.

Once the simulator has completed the task assignments (truck dispatching), the trucks embark on executing their specified tasks, involving a series of transportations between differing cranes. The completion of tasks sees the trucks either progressing to subsequent tasks, contingent upon their availability, or reverting to an idle state in the absence of pending tasks. The culmination of the simulation is marked by the successful execution of all assigned tasks, providing a comprehensive outlook on the operational workflow of the port.

Incorporating such a meticulously designed approach in the simulation unveils a wealth of nuanced insights and a multifaceted perspective of the intricate operations within container ports. It is an invaluable tool for the exhaustive evaluation, assessment, and ongoing refinement of varied truck dispatching strategies, ensuring they are continually optimized to mirror the dynamic nature of port operations.

In this thesis, every piece of experimental data subsequently presented is derived from the simulator depicted in Fig. 1.2. This simulator serves as a crucial tool in fostering an enriched environment that closely mirrors real-world conditions, thereby facilitating a meticulous evaluation and exhaustive testing of the various algorithms in question. The utilization of this sophisticated simulator not only enhances the credibility of the assessments but also provides invaluable insights into the nuanced interactions and inherent complexities of the simulated environment, mirroring the intricate dynamics of container port operations.

The rigorous testing environment created by the simulator enables a thorough examination of algorithmic functionalities, potential shortcomings, and areas of improvement. It acts as a conducive platform, allowing for the exploration and validation of theoretical constructs and algorithmic formulations in a controlled yet dynamic setting, reflecting the challenges and constraints found in actual port operations. By simulating real-world conditions with a high degree of accuracy, it provides a reliable basis for refining existing algorithms and developing innovative solutions designed to optimize container port logistics.

The inclusion of the simulator is pivotal for progressing algorithmic research, laying a solid foundation for delving deeper into the intricate layers of algorithmic structures and operational mechanisms. It acts as a catalyst, spurring advanced research endeavors and encouraging the exploration of novel methodologies and strategies, ultimately aiming to contribute significantly to optimizing logistical operations within container ports and to the broader field of operations research.

Furthermore, this thesis employs traditional event-based simulation to evaluate the performance of each algorithm, ensuring fairness in comparison. Additionally, in Chapter 6, we discuss the differences and limitations of traditional event-based simulation in contrast to real port environments and propose an optimized learning-based simulation method.

1.2.4 Baseline Manual Heuristic Truck Dispatching Method

Container port truck dispatching stands at the intersection of the global supply chain, acting as a linchpin that upholds the integrity and efficiency of seaport operations. However, this task is characterized by its complex, NP-hard nature, making optimal solutions elusive. Given the aim of this research—to contrast the performance of various truck dispatching methodologies—it becomes imperative to establish a coherent baseline for evaluation.

Historically, as depicted in Fig. 1.4, ports predominantly employed the Separate Truck Pool strategy with operator-driven operations. Within this framework, operators, armed with their experiential knowledge, continuously allocated trucks to pools associated with distinct QCs. A truck, post-task completion, would evaluate the task availability of its tethered QC. In scenarios with extant tasks, the truck would transition to the pertinent task location. Conversely, absent tasks would render the truck stationary, awaiting operator directives. This model, while seemingly simplistic, has been upheld by operators who, through years of on-ground experience, have managed to sustain remarkable operational efficiencies. However, a port's efficiency can oscillate based on the operator's proficiency—a novice operator can significantly hamper operational fluidity.

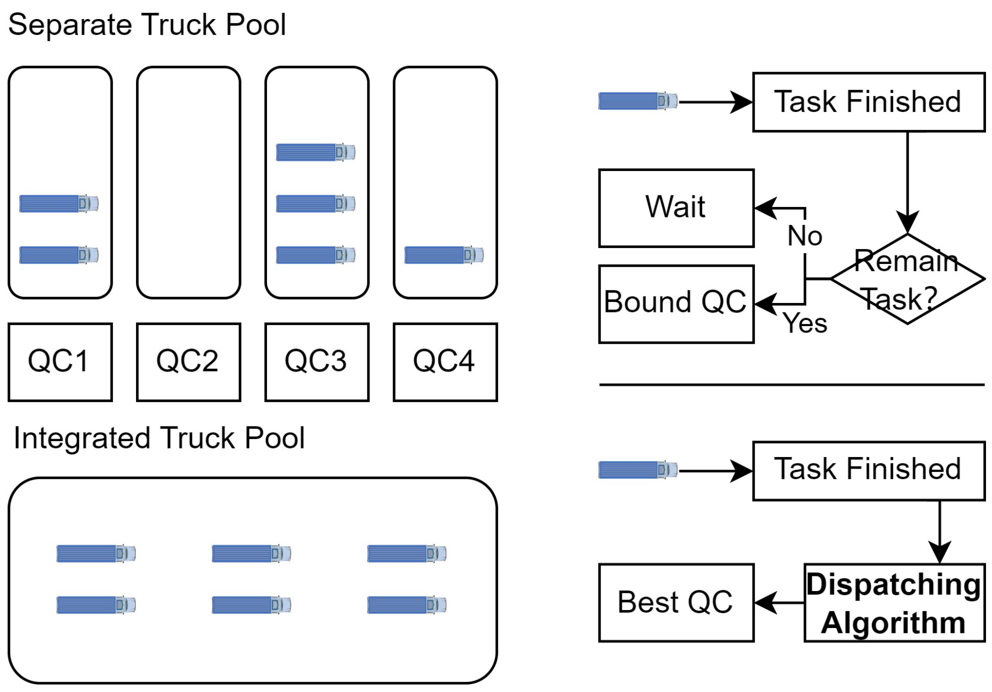


Figure 1.4: Separate Truck Pool vs. Integrated Truck Pool

In acknowledgment of the invaluable experiential reservoir that seasoned operators bring to the table, our research embarked on an expedition to structure this wisdom. Through a rigorous regimen of interviews, surveys, and questionnaires, Chen et al. (2016) abstracted the operational strategies into a manually crafted heuristic, detailed in Algorithm 1. This heuristic not only encapsulates the nuanced strategies honed by operators over time but also furnishes a robust benchmark for evaluating our innovative approaches.

At the algorithm's core lies an intricate lattice of parameters meticulously

Algorithm 1 Manual Heuristic Truck Dispatching Algorithm

```
Require: Parameters $parameter$, Travel Time $t$
  function $heuristic(QC, truck)$
      if $truck\_num < desired\_trucks$ then
          $score \leftarrow travel\_time * (truck\_num - priority)$
      else
          $score \leftarrow travel\_time * desired\_trucks$
      end if
      if $truck\_num \geq truck\_limit$ then
          $score \leftarrow score + 200000$
      end if
      return $score$
  end function
```

curated from operators' insights. Key among these are *desired_trucks*, indicative of the optimal truck count for a QC, *priority*, which demarcates Quay Crane (QC) hierarchies, and *truck_limit*, the maximal truck count per QC. These are real-time metrics, namely *truck_num*, which quantifies the trucks tethered to a QC, and *travel_time*, signifying the truck's transit latency to a QC's task source node. The value 200,000 serves as a penalty for any QC whose truck number exceeds the *truck_limit*. This figure was determined based on the experience of operators within the port company. Synthesized, these parameters and metrics forge a scoring algorithm for QCs.

When this manual heuristic is employed, the dynamic truck dispatching system automatically calculates a score for each potential task. It then evaluates the merits of each task based on this score and assigns the most suitable task to the vehicle. The process of calculating the score using the manual heuristic is detailed in Algorithm 1. It is important to note that there is a supporting dynamic truck dispatching system responsible for assigning tasks to trucks in addition to this algorithm. As this thesis primarily discusses the algorithmic aspect, it does not delve into the details of the dynamic truck dispatching system.

In an example like in Fig. 1.5, suppose there are two idle trucks, Truck 1 and Truck 2, and two QCs, QC 1 and QC 2, with tasks. Using a simple heuristic, where $score = powa * travel\_time$, 'powa' represents the number of trucks queued at a QC, and 'travel_time' is the time it takes for a truck to reach the corresponding crane. Assume that there are 1 and 2 trucks queued at QC 1 and QC 2, respectively. The travel times for Truck 1 to QC 1 and QC 2 are 3 and 1, respectively, and for Truck 2, the travel time to both QC 1 and QC 2 is 1. Therefore, when assigning a task to Truck 1, the calculated scores for QC 1 and QC 2 are 3 and 2, respectively. Since the score for QC 2 is lower, Truck 1 is dispatched to QC 2. For Truck 2, the scores for QC 1 and QC 2 are calculated as 1 and 2, respectively, so Truck 2 is dispatched to QC 1. This example clearly illustrates how the score serves as the basis for truck dispatch in dynamic truck dispatching.

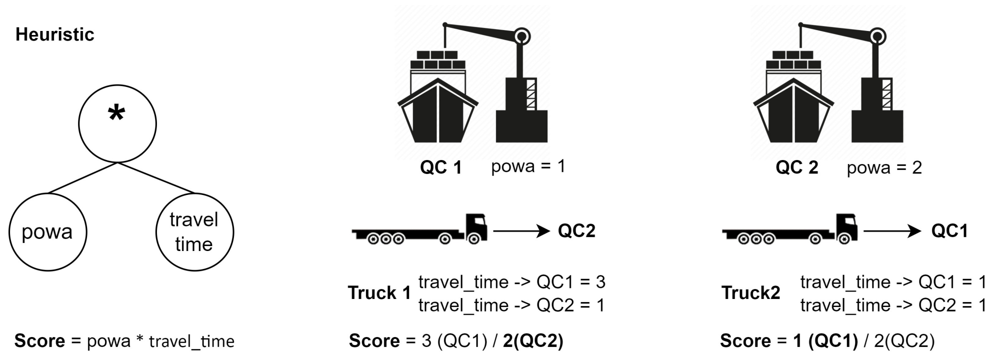


Figure 1.5: Example of Dynamic Truck Dispatching

As elucidated in Fig. 1.4, the advent of dispatching algorithms heralds a transformative approach to container port operations. With the embrace of the Integrated Truck Pool methodology, trucks are conglomerated into a singular pool. The primacy of manual operator interventions, which once required laborious one-by-one assignment of trucks to specified QCs, is rendered obsolete. In this new paradigm, the dispatching algorithm autonomously designates the truck to the most propitious QC for the succeeding task after completing a task. This operational shift offers a twofold advantage. First, it dramatically alleviates the operational burden on the coordinators, obviating the need for relentless manual interventions. Second, and more crucially, it bestows unparalleled stability to the entire terminal operation, ensuring that variations in operator experience and proficiency no longer wield the power to perturb the port's operational efficiency.

While the manually crafted heuristic, described earlier, represents an inaugural foray into algorithmic dispatching, it is not without its shortcomings. Predominantly, the heuristic's rigidity becomes palpable in several facets. The hyperparameters, which anchor the algorithm, might not invariably align with the optimal operational sweet spot. Moreover, the heuristic exhibits an inherent inertia when confronted with divergent operational environments, rendering it less adaptable. This inflexibility is further magnified in intricate operating scenarios where the heuristic's performance could waver, leading to inefficiencies like increased wait times, sub-optimal QC assignments, and potential operational bottlenecks.

Recognizing these limitations, this research endeavors to venture beyond the confines of rudimentary heuristics. The quest is to harness the prowess of machine learning, drawing upon its adaptability and predictive capabilities. By integrating diverse machine learning techniques, we aim to sculpt superior heuristic dispatching algorithms that rectify the inherent flaws of manual heuristics and resonate with the dynamism of varied port operational contexts. The intricacies of these methodological innovations and their empirical evaluations will be meticulously expounded upon in subsequent sections of this thesis.

1.3 Research Objectives and Scope

1.3.1 Objectives

The primary objective of this thesis is to enhance the efficiency and throughput of container port operations significantly. This enhancement is achieved by strategically applying advanced machine learning methods to optimize the truck dispatching process. By incorporating these technologies, the thesis aims to develop sophisticated algorithms capable of predicting container traffic patterns and optimizing truck movements. This approach is expected to reduce idle times and increase overall operational efficiency.

Further, the thesis explores the implementation of intelligent dispatching systems that dynamically adapt to the changing conditions of the port. These systems will use real-time data and predictive analytics to ensure the most suitable allocation of trucks to various tasks, thereby streamlining the port's operations. Integrating machine learning and metaheuristic in operational decision-making enables handling a higher volume of container traffic with greater accuracy and efficiency.

Central to this endeavor is the alignment of port operations with the rapidly expanding global demands for containerized shipping. The thesis aims to aid port companies in keeping pace with the evolving global container port sector and enhancing their economic effectiveness and profitability. By adopting these advanced methodologies, ports are expected to become more efficient and responsive to the complexities of global trade.

This approach enhances the role of ports as critical hubs in the global supply chain, contributing to smoother and more efficient logistics and trade flows. It positions ports at the forefront of innovation in logistics and transportation, setting new standards for operational efficiency, sustainability, and profitability in the industry. This comprehensive approach is instrumental in shaping a more streamlined, intelligent, and high-performing container port infrastructure ready to meet the challenges of modern global trade.

The key objectives of this thesis are articulated as follows:

- To optimize truck dispatching within container ports to minimize delays, reduce congestion, and maximize resource utilization efficiency. This optimization is crucial for streamlining port operations and enhancing overall productivity.
- To employ advanced machine learning and metaheuristic algorithms to facilitate automated decision-making in truck dispatching. This involves developing and implementing algorithms to efficiently process data and make informed decisions, reducing reliance on manual intervention.
- To transition truck dispatching from traditional, heuristic-based methods to innovative, data-driven strategies. This shift represents a move towards more scientifically grounded and technologically advanced approaches in managing port logistics.
- To improve the accuracy of port simulations, thereby enhancing operational planning, forecasting, and the effectiveness of algorithm evaluation. Accurate simulations are vital for testing and refining dispatching strategies, ensuring they are robust and applicable in real-world scenarios.

These goals aim to transform the operational efficiency of ports, making them pivotal nodes in the global trade network. The thesis explores integrating sophisticated machine learning techniques into dispatching systems, fostering a smarter and more adaptive port ecosystem. This research is poised to catalyze a paradigm shift in port operations, paving the way for a new era of intelligence and efficiency in global trade and commerce.

1.3.2 Scope

The scope of this thesis is anchored in the innovative application of machine learning and metaheuristic techniques specifically tailored to optimize truck dispatching within the dynamic environment of container ports. This research extensively explores the intricacies of real-time scheduling and the critical decision-making processes inherent in such settings. The thesis aims to significantly enhance operational efficiency, reduce bottlenecks, and streamline cargo flow by leveraging advanced computational methods.

Central to this exploration is a detailed examination of the challenges and opportunities presented by container port operations' fast-paced, often unpredictable nature. The study meticulously analyzes how machine learning can be applied to predict traffic patterns, optimize task assignments, and improve the overall management of resources. This approach contributes to reducing operational delays and plays a vital role in increasing the throughput and profitability of ports.

While the primary focus of the thesis is deeply rooted in container ports, its findings and methodologies have broader implications that transcend this specific context. The nature of the problems addressed — characterized by their need for swift, data-driven decision-making in constantly evolving scenarios — is common to many real-time dispatching problems across various sectors. Consequently, the insights gained from this research apply to other areas facing similar challenges, such as logistics networks, manufacturing processes, and urban traffic management.

In essence, the scope of this thesis not only sheds light on the complexities and potential solutions for container port operations but also paves the way for applying these advanced machine learning techniques in various other real-world contexts. The methodologies developed and lessons learned promise to revolutionize how real-time dispatching problems are approached and solved, offering far-reaching benefits beyond the confines of container port logistics.

The scope of this thesis encompasses several key aspects, each contributing to the overarching goal of enhancing operational logistics through advanced computational methods:

The following key aspects define the scope of this thesis:

- At its core, the thesis focuses on optimizing truck dispatching within container ports by leveraging machine learning and metaheuristic techniques. The aim is to substantially enhance the efficiency and effectiveness of decision-making processes in truck dispatching, ensuring smoother port operations.
- An in-depth exploration of real-time scheduling challenges is undertaken, particularly those prevalent in container port environments. This part of the research delves into the complexities and dynamics of making fast, strategic decisions under changing operational conditions, highlighting the need for adaptive and responsive solutions.
- The scope extends beyond container port operations, as the methodologies and algorithms developed have broader applicability. They

can be adapted to address similar dispatching and scheduling challenges across various sectors. This aspect of the research underscores the solutions' versatility and potential impact in diverse real-world contexts.

- This thesis also studies port simulation accuracy. This is crucial for validating the effectiveness of the dispatching strategies proposed. The research assesses the fidelity of simulations in mirroring real-world conditions and the reliability of the outcomes they produce, ensuring that the strategies are grounded in practical reality.

These areas collectively define the boundaries and focus of the thesis, framing a comprehensive approach to addressing specific and general challenges in operational logistics.

With its focused yet expansive approach, this thesis significantly contributes to operational logistics, offering versatile solutions for complex scheduling issues across various industries. The methodologies developed are particularly noteworthy for their ease of transferability to other real-world scheduling problems. This adaptability stems from the shared common features among these scheduling challenges, such as dynamic environments, the need for real-time decision-making, and the complexity of managing resources.

The core of this research lies in its ability to address the specific needs of container port dispatching and provide a template for solving analogous problems in different sectors. Whether it's managing logistics in manufacturing, coordinating tasks in large-scale construction projects, or optimizing routes in urban traffic systems, the principles and algorithms detailed in this thesis can be readily adapted. This adaptability ensures that the solutions proposed here are not confined to a single context but have broad applicability, poised to bring efficiency and innovation to various operational logistics scenarios.

1.4 Thesis Structure

This thesis is organized into several chapters, each elucidating specific aspects of the research undertaken.

- **Chapter 2: Literature Review and Background**
  This chapter provides an extensive review and background of the literature pertinent to container port logistics, dispatching strategies, and the application of machine learning in operational optimization.
- **Chapter 3: Genetic Programming in Dispatching Strategy Generating**
  This chapter delves into utilizing Genetic Programming for generating dispatching strategies, exploring its methodology and implications in optimizing port operations.
- **Chapter 4: Reinforcement Learning Assisted Method in Dispatching Strategy Generating**
  This chapter focuses on the application of Reinforcement Learning as an assistive method in generating dispatching strategies, elaborating on its operational dynamics and contribution to enhancing dispatching efficacy.
- **Chapter 5: Machine Learning Assisted Methods in Dispatching Strategy Evaluation Accuracy Enhancing**
  This chapter discusses the role of Machine Learning in improving the

accuracy of dispatching strategy evaluation, exploring various methods and their impact on evaluation precision.

- **Chapter 6: Neural Networks Assisted Methods in Dispatching Strategy Refinement and Evaluation Acceleration**
  The utilization of Neural Networks for refining and accelerating the evaluation of dispatching strategies is the central theme of this chapter, exploring its methodologies and implications in detail.

- **Chapter 7: Conclusion and Future Work**
  This final chapter synthesizes the research findings, draws conclusions from the investigated methodologies, and proposes avenues for future work in dispatching strategy optimization.

Chapter 2

Literature Review and Background

The epoch of international trade and commerce has been invariably intertwined with the port industry, serving as a pivotal hub for economic proliferation. In contemporary times, the mosaic of technological advancements coupled with the escalating imperatives of global commerce has precipitated the evolution of the **Intelligent Port and Port Optimization** paradigm. This paradigm, transcending mere operational enhancements, encapsulates a holistic transformation, integrating advanced technologies and innovative strategies to augment port operations' efficiency and resilience substantially. Within this expansive canvas, the **Container Port Truck Dispatching** remains a core focus, dictating the proficient circulation and allocation of container trucks. As the nerve center of port logistics, its optimization directly impacts ports' overall throughput and performance.

In pursuing refining these dispatching methodologies, academia and industry alike have focused on **Hyper-Heuristics**, offering a layered approach to problem-solving by amalgamating a repertoire of heuristics. This quest for perfection has also witnessed the integration of cutting-edge computational paradigms such as **Genetic Programming**, which mirrors nature's evolutionary algorithms to distill optimized dispatching strategies, and **Ensemble methods and Reinforcement Learning**, an adaptive paradigm empowering systems to navigate and adapt within multifaceted environments. And the fusion of GP with **Recurrent Neural Network and Transformer**. This combination capitalizes on GP's inherent evolutionary optimization strengths, RNN's aptitude for recognizing temporal sequences, and Transformer's self-attention mechanisms. Together, they offer a more refined dispatching strategy and significantly accelerate simulation speeds. Through this literature review, we journeyed to unravel these intertwined domains, elucidating their significance, intricacies, and potential confluence in sculpting the future of container port truck dispatching.

2.1 Intelligent Port and Port Optimization

In an era where the interdependence of global economies is increasingly pronounced, ports play a pivotal role as gateways to international trade. The surge in maritime traffic and the intricacies associated with burgeoning global supply chains have underscored the imperativeness of modernizing and streamlining port operations. Central to this paradigm shift is the concept of the *Intelligent Port*, a manifestation of the marriage between maritime logistics and cutting-edge technological innovations (de la Peña Zarzuelo et al., 2020; Wu et al., 2013).

At its core, an Intelligent Port is an ecosystem where traditional operational methodologies are augmented or replaced with advanced technologies such as the Internet of Things (IoT), artificial intelligence (AI), machine learning, and big data analytics (Xisong et al., 2013; Mi and Liu, 2022; Tsou, 2019). Collectively, these technologies aim to enhance operational efficiency, reduce logistical bottlenecks, increase throughput, bolster safety standards, and usher in a new era of sustainability and environmental responsibility in port operations.

The ramifications of Intelligent Ports extend beyond mere operational efficiency. Given the environmental concerns of our age, such ports, with their emphasis on cleaner and more efficient operations, are playing a pivotal role in reducing the carbon footprint of maritime activities, thus directly contributing to global sustainability goals (Fahdi et al., 2021; Clott and Hartman, 2013). Moreover, by leveraging real-time data analytics, Intelligent Ports enhance predictability and reliability, two facets that are of paramount importance in a world where just-in-time deliveries and lean supply chains dominate commercial imperatives (Indraratna et al., 2011; Filom et al., 2022).

However, transitioning to an Intelligent Port isn't solely about integrating smart devices or employing AI-driven solutions. At the heart of this transition is the overarching need to ensure these devices and solutions operate harmoniously, driving the port towards its strategic objectives. This harmonization is where *Port Optimization* becomes indispensable (Song et al., 2015). Port Optimization encompasses a plethora of strategies and methodologies aimed at ensuring every resource, be it human, machine, algorithm, or infrastructure, is utilized optimally. It's a multidisciplinary approach that draws from fields such as operations research, logistics, data science, and industrial engineering (Yang and Guo, 2020).

Over the decades, the domain of Port Optimization has witnessed significant evolution. From simplistic berth allocation strategies of yesteryears to today's sophisticated solutions that integrate real-time data from multiple sources to optimize every facet of port operations, including quay crane scheduling, yard management, truck dispatching, and even hinterland logistics (Heilig et al., 2020). This evolution has been driven by both necessities, given the growing complexity of port operations, and the availability of advanced technological solutions that have broadened the horizons of what's achievable.

The coordination between Intelligent Ports and Port Optimization is reshaping the maritime logistics landscape. As ports continue to play their crucial role in global trade, the emphasis on making them smarter, more efficient, and sustainable will only grow, driven by technological advancements and the relentless pursuit of operational excellence (Moros-Daza et al., 2020).

2.2 Container Port Truck Dispatching

Container port truck dispatching refers to the allocation and management of trucks that transport containers within the port premises (Choi et al., 2011). On the surface, this may seem like a mere logistical detail. However, delve deeper and grasp the gravity of its implications. An efficient dispatching system ensures that containers are promptly moved from ships to storage yards and vice versa, preventing costly ship idling at berths and ensuring swift vessel turnaround. Conversely, a lackluster dispatching system can lead to monumental bottlenecks, cascading ramifications on overall port efficiency, ship delays, congested storage areas, and escalating operational costs (Talley, 2006a).

The sheer volume of containers handled by modern ports further underscores the importance of optimized truck dispatching. With globalization driving a relentless increase in trade volumes, ports grapple with the daunting challenge of handling thousands of containers daily. Every minute a container remains unattended on the quay, or a truck idles waiting for its next task, translates into lost efficiency and economic ramifications.

Moreover, container truck dispatching is not just about the immediate movement of goods. It symbolizes a port's commitment to efficiency, innovation, and reputation in the more extensive supply chain network (Song, 2021). In a globalized world where timely delivery is paramount, ports with efficient truck dispatching systems are more likely to attract global shippers and carriers, ensuring their sustained relevance and competitiveness in international trade.

In essence, container port truck dispatching is much more than a logistical exercise. It is a strategic endeavor deeply intertwined with a port's operational efficacy, economic viability, and standing in the global maritime industry (Kuzmicz and Pesch, 2019). This literature review focuses on the nuances of this critical operation, elucidating its significance and the myriad methodologies devised over the years to refine and optimize it.

2.2.1 Port Operations and Optimization

Container ports stand as essential nodes in the global supply chain. They shoulder the complex responsibility of synchronizing many operations to ensure the uninterrupted flow of goods. The primary driving force behind these operations is optimization, with the goals of enhancing efficiency, minimizing downtime, and capitalizing on the fullest potential of both human and infrastructural resources.

The evaluation of port efficiency often leans on the metric TEU/h (Twenty-Foot Equivalent Units per hour). This metric serves a dual purpose. Firstly, it quantitatively measures a port's operational capability, indicating the number of standard-sized containers processed within a given hour. Secondly, it acts as a barometer for operational excellence. Elevated TEU/h values represent a port's proficiency in expeditiously managing voluminous traffic, reducing ship idle time, and facilitating prompt turnarounds. For port operators, this metric becomes a benchmark for performance assessment, while for potential clients, it offers insight into the port's reliability and efficiency.

Despite the common goal of optimization, the avenues to achieve this within a port are diverse:

Berth Optimization: As vessels arrive, the berth area becomes their primary docking point. The art of berth optimization lies in astutely allocating ships to docks, curtailing waiting periods, expediting loading and unloading processes, and averting potential congestion (Dai et al., 2008; Venturini et al., 2017; Ting et al., 2014; Golias et al., 2009; Arango et al., 2013). Key factors underpinning this optimization include ship dimensions, cargo priority, expected dock time, and equipment availability.

Yard Optimization: Containers find their temporary residence in specific yard zones after unloading. The essence of yard optimization is strategically positioning these containers, considering variables like cargo nature (for instance, perishable or hazardous), imminent retrieval dates, and subsequent transportation modes (such as road or rail) (Chen et al., 2004; Zhen, 2016; Chen et al., 2003; Sha et al., 2017; Zhen et al., 2013). Mastery in yard management translates to reduced retrieval durations and unhindered container accessibility.

Hinterland Logistics: This facet transcends the port's immediate confines and delves into the transit of goods from the port to their final inland destinations (Behdani et al., 2020; Rodrigue and Notteboom, 2009; Irannezhad et al., 2020; Bergqvist, 2012). It's a jigsaw puzzle, coordinating with diverse transportation mediums like railways and trucks, ensuring timely dispatch and receipt of commodities.

Gate Operations: As the port's entry and exit conduits, gate operations oversee the fluid movement of trucks and other vehicles. Their efficiency is paramount, as even minor hitches can trigger significant operational setbacks (Keceli, 2016; Chao and Lin, 2017; Lai and Leung, 2000). Streamlined gate operations necessitate rapid verification, thorough documentation, and unobstructed goods transit.

Amid this maze of operations, container trucks stand out as the linchpins. These vehicles seamlessly connect the disparate segments of the port, forming a bridge between berths, yards, and beyond. Their pivotal role cannot be understated—delays or inefficiencies in container truck movements can ripple across the entire operational chain, derailing the rhythm of the port. Thus, proficient truck dispatching is not just about transporting containers but is intricately tied to preserving the port's operational harmony.

In encapsulation, while every port shares the overarching ambition of optimization, the path to this goal demands an in-depth exploration of each operational segment, an understanding of its inherent challenges, and crafting strategies tailored to these unique requirements.

2.2.2 The Truck Dispatching Problem

Truck dispatching is a central pillar in port operations. Its role is akin to the nervous system in a living organism, transmitting vital signals across various parts to ensure harmony and fluidity. Maintaining a constant and streamlined movement of goods is at the core of its importance. This involves managing a multitude of trucks, each with their schedules, to ensure that the berth areas remain free of congestion and that the yard areas witness an uninterrupted flow of containers.

Historically, the initial attempts to organize this logistical ballet revolved around static methods (Abdelmagid et al., 2022). These methods, often characterized by predetermined schedules and operational parameters, were the early architects of order in a chaotic environment. These systems relied on pre-established routes, times, and sequences, assuming a more or less stable operating condition. On paper, they appeared to be the logical choice, transforming the seemingly anarchic movement of trucks into an orderly flow. The predictability they introduced was valuable, allowing ports to plan and allocate resources with a reasonable degree of confidence.

However, the inherent dynamism and unpredictability of port operations soon revealed the cracks in these static methodologies. Ports are buzzing hubs of activity where countless variables are constantly in flux. External factors, such as weather conditions, can delay ship arrivals. Operational challenges, like equipment breakdowns or sudden surges in container volumes due to peak trading seasons or geopolitical events, add another layer of unpredictability. In such an environment, relying solely on static methods meant disruptions often led to significant cascading delays, bottleneck formations, and resource wastage.

Recognizing the limitations of these traditional approaches, the industry saw a paradigm shift towards dynamic dispatching methodologies. Unlike their static counterparts, dynamic methods are inherently agile. They operate on real-time data, making on-the-spot decisions to accommodate the ever-shifting landscape of port operations. These systems can, for instance, immediately adjust truck dispatching patterns based on sudden changes in ship arrival times or prioritize certain containers based on real-time demand and urgency.

Furthermore, the advent of advanced technologies and data analytics has equipped these dynamic methods with the tools to not only respond to immediate changes but also to predict potential disruptions. This predictive capability, harnessed through machine learning algorithms and data-driven insights, has added another dimension to dynamic dispatching. By forecasting potential challenges, these methods can proactively adjust dispatching strategies, further enhancing the resilience and efficiency of port operations.

In conclusion, while the essence of the dispatching problem remains unchanged – ensuring the right truck is at the right place at the right time – the methods to achieve this have evolved considerably. In the modern era of smart ports and intelligent logistics, dynamic dispatching, empowered by real-time data and predictive analytics, is the beacon of efficiency and adaptability.

2.2.3 Challenges in Port Optimization

Container terminals are intricate hubs of logistical operations where countless variables converge to determine overall efficiency. While it's understandable that the attention has traditionally gravitated towards optimizing sea-side operations, particularly those associated with QCs, the ever-evolving dynamics of modern ports underscore the need for a holistic approach. This approach not only encompasses sea-side operations but also interlaces with the equally important land-side operations, where YCs and trucks play pivotal roles.

Vast Scale of Operations: Container terminals are sprawling infrastructures that handle millions of containers annually. This translates to thousands of daily operations, each with its unique set of parameters such as timings, routes, and priorities. Organizing these operations demands vast data processing, timely decision-making, and continuous monitoring. Moreover, as trade volumes continue to grow, the scale of operations only intensifies, exacerbating the complexity of optimization.

Interdependencies and Synchronicity: The terminal's workflow is much like a well-orchestrated symphony; every component must synchronize with the other to achieve optimal results. QCs, YCs, and trucks have interdependent roles. For instance, a delay in QC operations can trigger a domino effect, causing delays for trucks awaiting containers and YCs positioned for container storage or retrieval. These intricate interrelations mean that optimization in one area can inadvertently lead to complications in another if not carefully managed.

Non-Linearity Due to Equipment Limitations: Every piece of equipment, from cranes to trucks, has its operational limits. For example, a QC can only handle a specific number of containers per hour, and trucks have maximum load capacities. These limitations introduce non-linear constraints into the optimization problem. Simply put, enhancing the efficiency of one component doesn't guarantee a linear improvement in overall terminal performance. There's a threshold beyond which pushing equipment yields diminishing returns or even counterproductive outcomes.

Operational Uncertainties: The dynamic nature of port operations introduces multiple sources of unpredictability. Unforeseen equipment malfunctions, abrupt weather changes, or unscheduled ship arrivals can disrupt the best-laid plans. These uncertainties can rapidly cascade, turning minor disruptions into major operational challenges.

Balancing Priorities: At any given moment, a terminal might be juggling multiple priorities, from servicing mega-ships with tight schedules to managing storage for containers awaiting hinterland transportation. Striking a balance between immediate tasks and long-term operational goals is a recurring challenge.

Considering these multifaceted challenges, it becomes evident that optimization in container terminals isn't merely about enhancing a singular operation but orchestrating a complex dance of interrelated activities. Modern terminals, recognizing this, are increasingly leveraging data analytics, machine learning, and other technological advances to navigate these challenges and foster a more integrated optimization approach.

2.2.4 Findings in Port Optimization

Recent research has explored QC, YC, and truck efficiency. Works by Paul et al. (Schonfeld and Sharafeldien, 1985), Kim and Park (Kim and Park, 2004), and Kaveshgar et al. (Kaveshgar and Huynh, 2015) have shed light on optimizing QCs. These studies, while significant, have often overlooked truck availability, a factor that directly impacts QC efficiency. Conversely, research by Lai et al. (Lai and Lam, 1994), Zhang et al. (Zhang et al., 2003), and Chen et al. (Chen et al., 2007) has emphasized the efficiency of YCs. However, many of these studies operated under the unrealistic assumption of unlimited truck availability.

Further studies recognized the need to integrate QCs and YCs, with trucks as the crucial connectors. In this context, truck dispatching optimization becomes paramount, influencing not just local operations but overall port efficiency. Several methodologies have been proposed, including classical integer programming (Bish et al., 2007), nonlinear programming (Lu and Jeng, 2006), and heuristic algorithms (Bish et al., 2007), reporting substantial improvements in various performance metrics.

Nevertheless, real-world port operations, with variable QC and YC operational times, unpredictable truck speeds, and changing task sequences, introduce significant stochasticity. Offline optimization methods, despite their success, often falter when faced with such uncertainties. Although some algorithms (He et al., 2019; Zhen et al., 2011; Sislioglu et al., 2019) exhibit limited tolerance to unpredictability, they struggle when the uncertainties escalate. A noteworthy approach by a previous study (Chen et al., 2016) introduced an `online` heuristic-based truck dispatching methodology. While offering reasonable solutions, it lacked optimal guarantees (Pardalos and Romeijn, 2002) and adaptability across diverse scenarios, highlighting the need for advanced, adaptive hyper-heuristics.

Even though truck dispatching represents just a fraction of the entire container port optimization process, its impact permeates every corner of port operations. As modern ports transition towards intelligent systems, building a robust and efficient truck dispatching mechanism becomes paramount. This emphasis on truck dispatching arises due to its central role in synchronizing various port functions, from berth assignments to yard operations.

Any inefficiencies or delays in truck movements can ripple through the entire system, causing holdups and increased operational costs.

Moreover, as the complexity and unpredictability of port operations intensify, traditional methods often fall short of ensuring optimal dispatching. Introducing machine learning methods offers a promising solution. These methods can adapt to the dynamic nature of port environments, learn from historical data, predict potential disruptions, and make real-time decisions. Leveraging machine learning not only aids in handling the present operational challenges but also positions ports to be future-ready, able to scale and adapt as trade volumes and complexities evolve. Thus, refining and enhancing truck dispatching via machine learning becomes an imperative focus in the pursuit of creating a truly modern and intelligent container port.

2.3 Dynamic Dispatching Technologies

2.3.1 Hyper-Heuristics

The advent of hyper-heuristics has brought forth a promising new horizon in the realm of search and optimization methodologies (Burke et al., 2003). From their inception as a general-purpose, rapid prototype strategy, hyper-heuristics have shown remarkable potential in adapting to various problem domains, offering solutions of acceptable quality (Cowling et al., 2002; Pillay and Qu, 2018).

Defining the Hyper-heuristic Paradigm

Traditionally, heuristic methodologies were developed with a single-minded focus, targeting specific problems or a narrow range. Hyper-heuristics, however, venture beyond this narrow scope, aiming for a broader application. Instead of directly searching the solution space, as conventional or meta-heuristics do, hyper-heuristics operate in the heuristic space (Swan et al., 2014). The fundamental hypothesis is that the heuristic space is less tethered to specific problems than the solution space, enabling the creation of a more universal search mechanism. This can be understood as `heuristics to select or generate heuristics`, making the approach substantially different and more generalized than traditional algorithms.

Yet, while this approach seems like a pursuit for the universal optimizer, it's essential to remember the No Free Lunch theorem (Wolpert and Macready, 1997). This theorem dictates that no singular method can optimally address every conceivable problem. However, it leaves room for adaptive strategies to manage problems with overlapping features or structures. This theoretical allowance provides the philosophical foundation for hyper-heuristics. By employing various learning mechanisms—both online and offline—hyper-heuristics aim to bolster the universality of algorithms across different problems and scenarios.

The Structure of Hyper-heuristic

As delineated in Fig. 2.1, hyper-heuristics are based on a two-layered structure. For the traditional hyper-heuristic framework, the high-level heuristic, representing the first layer, does not grapple with problems directly. Instead, it plays the role of a maestro, either selecting from a pre-existing

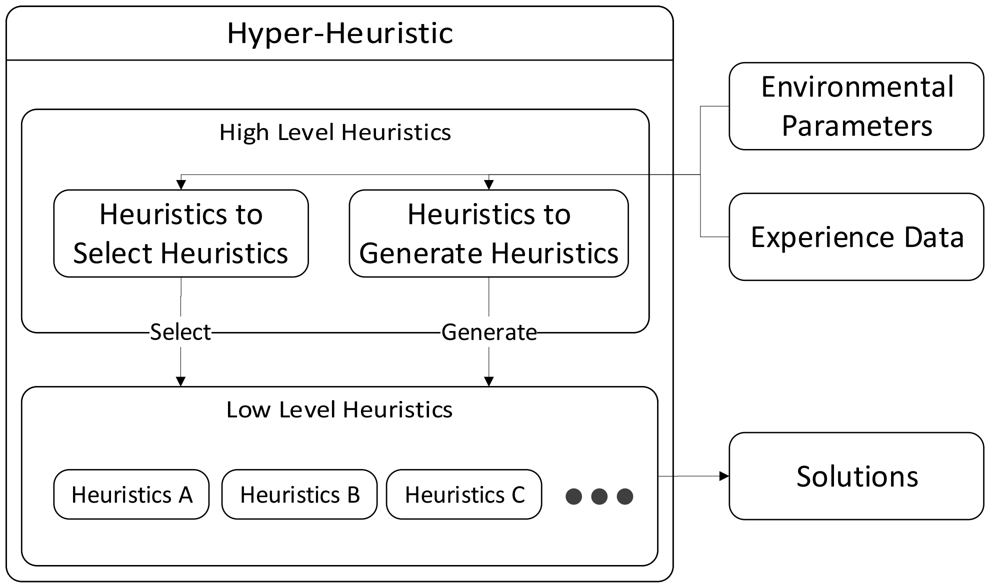


Figure 2.1: Hyper-heuristic Typical Structure

ensemble of heuristics or innovating new ones as the situation demands. Throughout this process, the high-level heuristic continually gleans from accumulated experiential data, refining its choices or creations for low-level heuristics. This methodological framework, while seemingly simple, provides a versatile platform for addressing a myriad of complex problems, as evidenced by its success in domains like educational timetabling (Hermansyah and Muklason, 2020; Bai et al., 2006), two-dimensional strip packing (Burke et al., 2010b; Domović et al., 2019), and vehicle routing (Qin et al., 2021; Chen et al., 2018a), to name a few.

In this thesis, the classification of hyper-heuristic approaches follows the model proposed by Burke et al. (2010a), as depicted in Fig. 2.2, which we reproduce here for completeness. This classification is based on two dimensions: (i) the nature of the heuristics' search space and (ii) the sources of feedback information.

Regarding the nature of the search space, there are two primary categories: (i) heuristic selection, which involves choosing or selecting from existing heuristics, and (ii) heuristic generation, which focuses on creating new heuristics by combining components of existing ones. Further, this dimension distinguishes between constructive and perturbative search paradigms, as explained by Hoos and Stützle (2004). Perturbative methods modify complete candidate solutions by changing one or more of their components. In contrast, constructive methods work with partial candidate solutions, iteratively extending them by adding missing components.

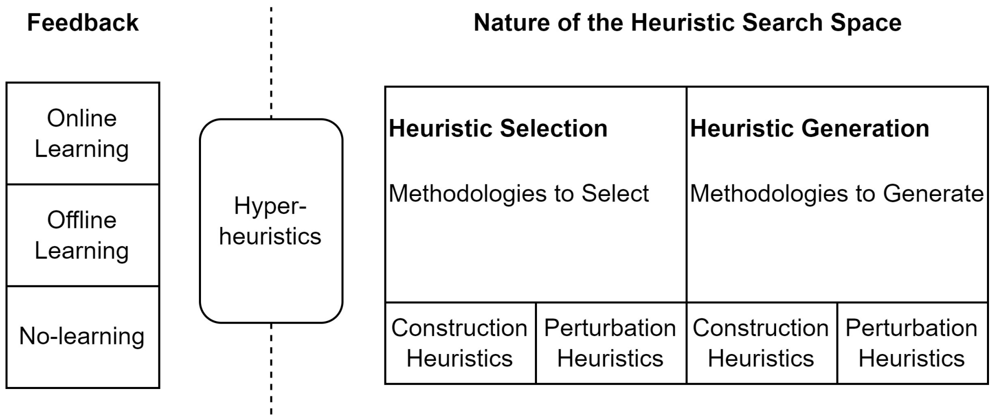


Figure 2.2: Hyper-heuristic Classification

A hyper-heuristic is classified as a learning algorithm when it incorporates feedback from the search process. We distinguish between online and offline learning based on the source of feedback. Online learning hyper-heuristics acquire knowledge during the problem-solving process. In contrast, offline learning hyper-heuristics gather rules or programs from training instances to apply to new, unseen instances.

These categories align with current research trends, yet it's noteworthy that some methodologies transcend these classifications. For example, hybrid methodologies that merge constructive with perturbation heuristics (Garrido and Riff, 2010; Elshaikh et al., 2016) or heuristic selection with heuristic generation (Zhao et al., 2021; Maturana et al., 2010; Remde et al., 2012) are increasingly prevalent, reflecting the evolving nature of this field.

As for selection hyper-heuristics, Drake et al. (2020) has extended the classification of selection hyper-heuristics in this research, which builds upon Burke et al. (2010a). This classification encompasses several critical dimensions:

1. **Nature of Feedback Received:** This category distinguishes hyper-heuristics based on their learning mechanisms, identifying those with online learning, offline learning, mixed learning methods, and those without any learning mechanism.

2. **Nature of the Low-Level Heuristics:** This aspect evaluates the variety and organization of low-level heuristics within the hyper-heuristic framework, considering their types, the set they form, and how they are grouped.

3. **Nature of Solutions:** This dimension focuses on the types of solutions employed by hyper-heuristics, whether they are single-point or population-based.

4. **Nature of Objective:** This dimension addresses the objectives targeted by hyper-heuristics, differentiating between single-objective and multiobjective approaches.

5. **Nature of Move Acceptance:** This section delves into the decision-making strategies used by hyper-heuristics for move acceptance, classifying these methods as either stochastic (probabilistic) or non-stochastic.

6. **Nature of Parameter Setting:** The final dimension examines how hyper-heuristics manage parameters, detailing approaches that are static, dynamic, adaptive, or self-adaptive.

Each dimension contributes significantly to defining the operational characteristics of selection hyper-heuristics, providing insight into their complex problem-solving capabilities.

Despite the growing prosperity and diverse applications of hyper-heuristics (Ryser-Welch and Miller, 2014), a significant gap remains in their application to truck dispatching within marine container terminals. The literature predominantly showcases studies focusing on selective hyper-heuristics, leaving generative hyper-heuristics, especially those adept at managing multi-scenario dynamics in truck dispatching, largely unexplored. This oversight presents a unique and promising opportunity for research and development. By leveraging advanced hyper-heuristic methodologies, there is substantial potential to optimize truck dispatching processes in marine container terminals, addressing complex operational challenges with innovative solutions.

Broadening the Horizons with Ensemble Approaches

Further contributing to the dynamism of hyper-heuristics is their ability to be employed in ensemble settings. In this thesis, we harness the power of GP as a hyper-heuristic generator of low-level heuristics while leveraging reinforcement learning as a mechanism for selecting and amalgamating GP individuals. This dual employment distinguishes our approach from standard hyper-heuristics, where the primary focus is pinpointing a singular heuristic aligning with the problem's predominant traits.

By proposing an ensemble framework powered by reinforcement learning, our strategy fosters collaboration amongst multiple low-level heuristics. This collective approach offers an innovative response to intricate multi-scenario challenges, frequently plagued by slow learning, challenging generalizations, and less-than-optimal performance. The underpinnings of this ensemble approach and its ramifications for the field will be explored in subsequent sections.

As this literature review elucidates, hyper-heuristics, emphasizing versatility and adaptability, offer an exciting frontier for optimization research. Their two-tiered architecture and potential ensemble approaches set the stage for tackling complex, real-world problems with renewed vigor and creativity. This thesis aims to contribute to this burgeoning field, focusing on applying hyper-heuristics in the nuanced domain of container port logistics.

2.3.2 Genetic Programming

Genetic Programming is a distinctive and widely acknowledged computational method in evolutionary computation. Pioneered by Fogel et al. (Fogel et al., 1966) and later popularized by Koza in the 1990s (Koza, 1994), GP capitalizes on principles of natural evolution, specifically mutation, crossover, and selection to evolve a population of programs, predominantly represented as GP trees. Over successive generations, these trees adapt and improve their capability to address complex problems through selection, crossover, mutation, and replacement mechanisms.

One of the standout capabilities of GP is its adaptability in solving a myriad of engineering and optimization issues. The technique has been leveraged in various applications, including optimizing autonomous vehicles control (Ardeh et al., 2022), predicting financial market trends (Christodoulaki et al., 2022), classifying images (Fan et al., 2022), symbolic regression (Chen et al., 2018b), and, pertinent to this study, dynamic truck dispatching (Chen et al., 2019). Specifically, in the container terminal simulation context, GP has displayed a commendable ability to discern intersection passing rules even without precise GPS data, underscoring its ability to infer patterns and behaviors from existing data sets (Elhenawy et al., 2014).

When juxtaposed with other prominent methods, such as decision trees, logistic regression, support vector machines, and artificial neural networks, GP distinguishes itself with three salient advantages. It champions flexible representations, which facilitate encoding intricate problem structures (Fan et al., 2023). Coupled with this, its powerful search algorithms make traversing expansive solution spaces feasible (Mei et al., 2022). Significantly, GP-derived heuristics are not just efficient in execution but also proffer partial interpretability, thus bolstering their utility in practical scenarios (Nguyen et al., 2017).

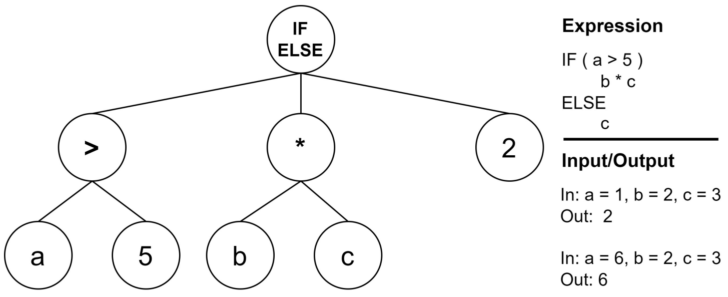


Figure 2.3: Genetic Programming Tree Structure

Traditional GP representations, such as the tree-based GP (Fig 2.3), have garnered favor due to their clear visualization and interpretability. Such structures facilitate the encoding, evolution, and evaluation of mathematical expressions with ease. While alternative non-tree structures like linear GP (Brameier and Banzhaf, 2007) and stack-based GP (Perkis, 1994) have their merits, the tree-based approach has been shown to represent decision-making logic, especially in multifaceted scenarios succinctly. This lucidity in representation can be instrumental when liaising with real-life decision-makers, such as port operators.

4

Moreover, the past decade has witnessed innovations to augment GP performance, encompassing methods such as double-layer GP (Chen et al., 2022), neural-network-assisted GP (Chen et al., 2023), and reinforcement learning-assisted GP (Yi et al., 2022). Although these techniques have made significant strides, one consistent challenge in GP-based approaches remains their computational intensity, especially when running extensive simulations to evaluate individual fitness within the GP population (Hildebrandt and Branke, 2015).

This computational demand highlights the importance of seeking more efficient and scalable GP-based methodologies, especially as real-world optimization problems grow in complexity and scale. The challenge is to strike a balance between computational efficiency and solution accuracy. A pivotal issue within the GP literature pertains to the high computational costs of continuously evaluating a rapidly expanding population of candidate solutions. Each individual in the GP population typically requires a detailed simulation or complex function evaluation to ascertain their fitness. These evaluations can become exceedingly time-intensive when operating in high-dimensional spaces or seeking solutions for intricate problems.

However, the advancements in parallel processing, cloud computing, and algorithmic improvements present promising avenues to mitigate these computational challenges. Additionally, cooperative coevolutionary GP (Nguyen et al., 2013) has emerged as a compelling methodology by segmenting problems into smaller, more tractable sub-problems, enabling concurrent evaluations and speeding up the evolutionary process. Such stratagems exemplify the continuous push in GP research to enhance the method's efficacy without compromising the solutions' quality.

It's also worth noting that the adaptability of GP provides an unparalleled opportunity to incorporate domain-specific knowledge, should it be available. Incorporating such knowledge can expedite the GP evolutionary process and lead to solutions more congruent with real-world constraints and expectations. Several studies have exemplified this approach, utilizing domain insights to guide the GP evolution or to prune the solution space effectively.

Furthermore, the literature reveals a growing interest in hybrid models that combine the strengths of GP with other optimization and machine learning techniques. For instance, integrating neural networks with GP (Chen et al., 2023) has resulted in models that leverage GP's adaptability and symbolic reasoning with the pattern recognition strengths of neural networks. Such hybrid models symbolize the future direction of GP research, wherein the method is seen not as an isolated tool but as part of a holistic, integrated toolkit for solving complex optimization problems.

In conclusion, the trajectory of GP in academic literature underscores its significance and versatility in addressing a wide array of optimization challenges. As computational resources evolve and methodologies mature, GP stands poised to play an even more influential role in shaping solutions for tomorrow's intricate real-world challenges. As researchers and practitioners, our charge is to continue probing the boundaries of GP, integrating it with emerging techniques, and harnessing its full potential to realize optimal, efficient, and interpretable solutions across diverse domains, especially in complex real-world optimization problems like dynamic container port truck dispatching.

2.3.3 Ensemble Methods

Ensemble methods, also known as multi-expert models, have attracted significant attention in recent years due to their ability to address complex problems by combining the strengths of several individual models or heuristics. These models are based on the premise that integrating the outputs of multiple experts can result in a more robust and accurate decision-making process, as opposed to relying on a single expert or model.

Ensemble methods have been widely employed in machine learning and pattern recognition tasks, where several base classifiers are combined to improve classification accuracy (Kuncheva, 2014). These techniques aim to create diverse and complementary classifiers by manipulating the training data or learning algorithms. Research has shown that ensembles can significantly improve algorithm performance, mainly when each algorithm exhibits different strengths and weaknesses (Polikar, 2006; Hong et al., 2024).

Ensemble methods have been employed in optimization to solve complex problems more effectively. For instance, the cooperative co-evolutionary algorithm (Ma et al., 2018; Qin et al., 2022; Theodorakos et al., 2022) divides a problem into smaller subproblems, which individual experts solve. The solutions are then combined to form a global solution. Similarly, the evolutionary forest (Zhang et al., 2021a) and particle swarm optimization (Ecer et al., 2020) apply multiple search strategies simultaneously, allowing the exploration of the solution space more effectively.

Leveraging the benefits of both ensemble methods and hyper-heuristics models, this paper introduces a novel learning-assisted heuristic ensemble model, which, to our knowledge, is the first to propose the utilization of reinforcement learning as a gate network for selecting/combining multiple low-level heuristics to address intricate multi-scenario problems. The learning-assisted ensemble structure in this paper is depicted in Fig. 2.4. The heuristics compete to generate output, while the gating network or-

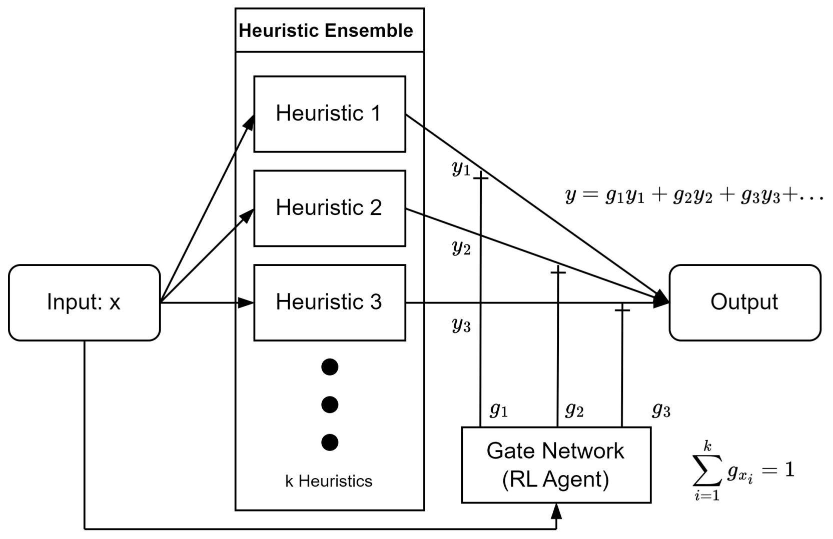


Figure 2.4: Learning-Assisted Heuristic Ensemble Framework

chestrates this contest. For each input x, the gating network acquires information about the performance of all heuristics $(y_1, y_2, y_3, ...)$ involved in addressing the task, and the output of each heuristic is compared with the target output y. The gating weights of heuristics $(g_1, g_2, g_3, ...)$ are adjusted based on the relative performance of that heuristic, compared to the other heuristics, for the specific input pattern. Within this framework, each heuristic is required to solve the problem solely within its designated area of focus, thus mitigating the model's complexity and training costs while enhancing the robustness of the generated solutions.

We posit that the learning-assisted ensemble hyper-heuristics model can deliver enhanced performance characterized by rapid convergence and superior generalization capabilities. Employing a weighted sum in this model facilitates the effective integration of individual heuristics' strengths, culminating in a more robust and powerful decision-making process. Moreover, including a gated network in the model expedites training by selectively concentrating on pertinent heuristics and discarding less effective ones, thereby streamlining the learning process and augmenting overall performance. This model is examined and analyzed on a complex multi-scene port truck dispatching problem, which involves uncertainty, in the following sections.

2.3.4 Recurrent Neural Network and Transformer

In the vast landscape of machine learning, numerous techniques have emerged to tackle intricate problems within transportation systems. The Recurrent Neural Network (RNN) and the Transformer model stand out due to their unique attributes and compelling use-cases.

Recurrent Neural Networks (RNN)

Recurrent Neural Networks (RNNs) have risen to prominence in the domain of time-series analysis due to their inherent ability to capture temporal dynamics and sequential dependencies. This is primarily due to the RNN's architecture, which allows for feedback loops, making it particularly apt for time-series prediction tasks. In logistics, RNNs have demonstrated significant prowess in demand forecasting (Zhao et al., 2020) and vehicle routing (Xu et al., 2021). By leveraging the capacity of RNNs to remember past information, these networks are ideally suited to address problems where temporal patterns and sequences are crucial.

Transformers

On the other side of the spectrum, the Transformer model, introduced by Vaswani et al. in 2017 (Vaswani et al., 2017), has revolutionized various domains, from natural language processing (Wolf et al., 2020b) and image processing (Chen et al., 2021) to data prediction (Wang et al., 2022).

Unlike the sequential nature of RNNs, the Transformer model leverages self-attention mechanisms, allowing it to weigh the relevance of different parts of an input sequence irrespective of their order. This inherent characteristic makes the Transformer model exceptionally effective in handling complex relationships and dependencies in data.

RNN, Transformer, and Genetic Programming

While both RNN and Transformer models offer unique advantages, GP has also solidified its place in optimization. Notably, GP has been employed for evolving dispatching rules in manufacturing systems (Zhang et al., 2023) and has found applications in routing within transportation networks (Ahvanooey et al., 2019), as well as in pattern recognition tasks such as medical text classifications (Cui et al., 2019; Liu et al., 2020). However, with its global search capabilities, GP can sometimes lead to suboptimal solutions due to its limitations in local search (Khayyam et al., 2020).

Recent initiatives have combined RNN and GP to address this gap, focusing on symbolic regression problems (Mundhenk et al., 2021), yielding promising results. This amalgamation leverages the RNN's temporal sequencing capabilities with GP's optimization strengths, creating a synergistic approach that provides more holistic and efficient solutions.

Moreover, a compelling case emerges for integrating the Transformer model into this framework. As container port truck dispatching exhibits intricate dynamics, the ability of the Transformer model to capture complex relationships can be instrumental. Furthermore, by employing the Transformer model as a surrogate, the computational efficiency of the system can be significantly enhanced, thereby boosting simulation speeds. As empirical evidence suggests, traditional surrogate models may not adequately handle the nuances of port vehicle assignment, emphasizing the need for more advanced models like the Transformer.

While RNNs, Transformers, and GPs have individually shown potential in various domains, their combined strength, especially in dynamic truck dispatching, offers a promising avenue for future research and applications. By harnessing their complementary attributes, we can refine dispatching strategies and significantly boost simulation speeds, resulting in more efficient and robust solutions for port operations.

2.3.5 Reinforcement Learning

Reinforcement Learning is traced back to the 1950s and has gained significant momentum with deep Q-learning introduced by DeepMind (Mnih et al., 2013). By facilitating agents to learn from their environment by interacting with it, reinforcement learning (Reinforcement Learning (RL)) operates on the principle of rewards and punishments, ensuring optimal decision-making. The vast potential of deep reinforcement learning has been harnessed across numerous fields, from operations research to self-driving technologies (Hubbs et al., 2020; Emuna et al., 2020).

However, despite its vast potential, the application of deep learning, particularly in real-life port operations, has been somewhat limited. Challenges such as training feasibility, convergence issues, and the intricate nature of sparse rewards, especially in dynamic port dispatching scenarios, have limited its widespread adoption (Khorasgani et al., 2020).

Deep reinforcement learning-based hyper-heuristics (DRL-HH) was introduced to counter these challenges, amalgamating principles with heuristic algorithms (Zhang et al., 2022; de Santiago Junior et al., 2020). Although it showed promise, most proposed methods rely on variants of a single heuristic. This implies that each time the reinforcement learning agent decides, it can only choose one heuristic as its action.

Considering the above context, a compelling argument arises for using RL as a gate network in ensemble models (Song et al., 2023). With its inherent capability of understanding and learning the environment and making decisions based on accumulated knowledge, RL can effectively evaluate the performance of individual heuristics in real time and adjust the gating weights accordingly (Lee et al., 2021). Such dynamic adaptability can be invaluable, particularly in intricate multi-scenario real-world challenges, allowing the model to focus on pertinent heuristics and discard less effective ones. This streamlines the decision-making process and potentially augments solutions' overall efficiency and robustness.

In conclusion, combining ensemble methods' strengths and adaptability to reinforcement learning offers a promising pathway for tackling complex optimization problems in various real-world scenarios. As we continue to advance in this domain, combining these methods promises to redefine our problem-solving capabilities.

2.4 Summary

The monumental intertwining of international commerce with the port industry has borne witness to several transformative phases, each redefining the core operational tenets of the sector. The dawn of the **Intelligent Port and Port Optimization** paradigm stands as a testament to this evolution, marking a transition from mere operational adjustments to a comprehensive overhaul powered by technology and innovation. At the heart of this paradigm, the intricacies of **Container Port Truck Dispatching** remain central, underpinning the essence of port logistics. The repercussions of its optimization cascade through the entire port infrastructure, determining its overall efficacy and performance.

Our exploration through the annals of academic and industrial pursuits in this domain has unveiled a rich tapestry of methodologies, each contributing a unique facet to the dispatching optimization problem. The layered sophistication of **Hyper-Heuristics**, the nature-inspired algorithms of **Genetic Programming**, the adaptive prowess of **Ensemble methods and Reinforcement Learning**, and the combination of GP with the **Recurrent Neural Network and Transformer** have all showcased their distinct advantages and challenges. The amalgamation of these techniques, each complementing the other, can redefine the landscape of dispatching strategies, infusing them with greater accuracy, adaptability, and speed.

As we conclude this literature review, it becomes evident that the optimization horizons in the container port truck dispatching domain are vast and ever-evolving. The convergence of classical heuristics with modern computational paradigms has ushered in a new era of possibilities, setting the stage for groundbreaking advancements in the field. With this foundation, we now transition into a deeper exploration, commencing with the intricate dynamics of machine learning in container port truck dispatching, spotlighting the role of Genetic Programming (GP) in generating truck dispatching strategies.

Chapter 3

Genetic Programming in Dispatching Strategy Generating

Truck dispatching, especially in the intricate environment of marine container terminals, is inherently laden with complex scenarios. One core observation is the varied impact of problem parameters or features across different operational scenarios. To elucidate, parameters such as the average quay crane load time and average quay crane unload time reveal their paramount influence in scenarios predominantly governed by loading and unloading operations, respectively. This inherent variability poses challenges to certain heuristic methods. While traditional Arithmetic Genetic Programming (Arithmetic Genetic Programming (AGP)) provides significant advantages over manual heuristics, such as improved adaptability, efficiency, and solution quality, it still harbors certain limitations in addressing dynamic scenarios.

In the broader scientific community, Genetic Programming with logic operators (Logic Genetic Programming (LGP)) often emerges as a favored choice to overcome these challenges. However, it is imperative to acknowledge that incorporating additional operators can inadvertently inflate the search space (Ebner, 1999). Such an expansion, while intending to enhance flexibility, may conversely lead to prolonged search times and often culminate in sub-optimal solutions attributable to deficient convergence qualities.

Despite the aforementioned enhancements brought by LGP over AGP, primarily due to its integration of logic operators, there remains a palpable challenge. The burgeoning search space, coupled with convergence issues in certain scenarios, even when the functional expression capability is fully leveraged, accentuates the need for a more refined approach.

This chapter seeks to bridge this gap. We present an exploration into a hierarchical genetic programming encoding structure tailored to harness maximally the strengths of logic operators within the GP framework. A pivotal aim is to circumvent the potential pitfalls of an exponentially growing search space. Anchored in inspirations from pioneering research on cooperative coevolution GP and innovative GP representations, we unveil the Cooperative Double-Layer GP Hyper-heuristic (Cooperative Double-Layer Genetic Programming Hyper-heuristic (CD-GPHH)). This novel framework bifurcates scenario grouping and dispatch ordering into distinct subpopulations. We posit that such a demarcation can augment the readability and elevate the solution quality in confronting the multifaceted, dynamic vehicle scheduling quandaries.

Delving deeper into the proposed CD-GPHH, we segregate GP individuals into dual cooperative strata: the high-scenario and normal-calculation layers. These layers operate with autonomy, possessing their distinct mutation and crossover paradigms. Notably, the high-scenario layer plays a pivotal role in determining the appropriate normal-calculation individual to be invoked for solution generation within a specific scenario.

The overarching ambition of this chapter is to forge a robust GP methodology. Through its lens, we aim to evolve dynamic truck dispatching heuristics of superlative quality seamlessly. Such evolved heuristics are poised to empower port companies, facilitating instantaneous decision-making in real-time and thereby adeptly maneuvering through the multifarious scenarios rife with uncertainties. Our empirical investigations span a gamut, encompassing the AGP, LGP, and the novel CD-GPHH. These were rigorously tested on rudimentary function fitting problems and the intricate real-world marine container terminal truck dispatching challenges. Our endeavors culminate in dual seminal contributions. Firstly, we underscore the potency of GPHH in unraveling real-world online combinatorial optimization challenges reminiscent of those pervading many expansive container ports. Secondly, our research brings a pioneering bi-level solution framework poised to inherently leverage the intricate structures of scenario switches, a hallmark of myriad real-world challenges, thereby enhancing the previously propounded GPHH.

To elucidate further, the ensuing sections of this chapter are structured methodically. Section 3.1, 3.2 and 3.3 delves deep into the intricate details of AGP, LGP, and our proposed CD-GPHH algorithm. Thereafter, Section 3.4 unveils the experimental design and its concomitant results, accompanied by a meticulous analysis. Concluding this chapter, Section 3.5 encapsulates the core insights, concurrently paving the way for prospective research avenues.

3.1 Arithmetic Genetic Programming

Arithmetic Genetic Programming is a seminal methodology within the broader ambit of GP. It harnesses the raw power of arithmetic operations to sculpt solutions to complex optimization and symbolic regression tasks. Grounded in the foundations of mathematics, AGP embeds an inherent capability to represent and tackle intricate problems, making it especially relevant for the task at hand: container port truck dispatching.

AGP operates by constructing mathematical expressions, often in tree forms shown in Fig. 3.1, that best describe a given problem or dataset. In truck dispatching, this translates to evolving expressions that optimize dispatch strategies based on myriad parameters, spanning from truck queues to QC operation times. Notably, this representation granularity differentiates AGP from manually crafted heuristics, which tend to offer a more generic outlook.

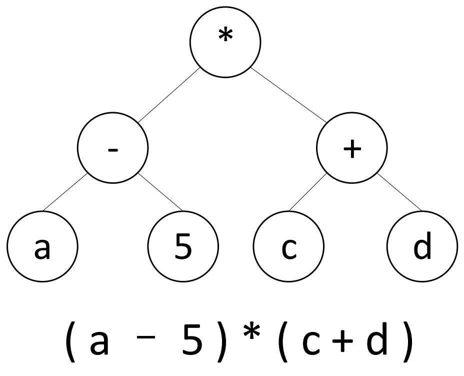


Figure 3.1: AGP Tree Structure

The core operation of AGP is shown in Algorithm 2, which involves initializing a population of random arithmetic expressions. Each of these expressions, termed 'individuals,' represents a potential solution to the problem. These solutions' efficacy or 'fitness' is then gauged against a predefined objective function, as mentioned in equation (1.4). The most fitting solutions are identified, and genetic operations—crossover, mutation, and reproduction—are applied, ushering in a new generation of evolved solutions. With each iteration, the solutions evolve, becoming more attuned to the underlying problem, leveraging the arithmetic operations and structures inherent in AGP.

Algorithm 2 AGP, LGP, and CD-GPHH Evolution Algorithm

```
Require: Initial Parameters $initial$
  $p \leftarrow NewPopulation$
  $p.initial\_individuals(initial.population\_size)$
  $generation \leftarrow 0$
  while $generation < initial.max\_generation$ do
      $p.calculate\_fitness()$
      $p.penalize\_long\_individuals()$
      $next\_generation \leftarrow NewPopulation$
      while $next\_generation.size() < p.size()$ do
          Insert an $individual$ to $next\_generation$ by
          Crossover, Mutation, or Reproduction in $p$
      end while
      $p \leftarrow next\_generation$
      $generation \leftarrow generation + 1$
  end while
```

To ensure an exhaustive exploration of the solution space and prevent premature convergence to sub-optimal solutions, AGP employs non-elitist tournament selection. This strategic choice amplifies genetic diversity in the population, paving the way for a more varied and robust solution. The iterative, evolutionary process perpetuates until a predefined criterion, such as a maximum number of generations or a fitness threshold, is met.

In the following subsections, we provide a comprehensive exploration of the inner workings of AGP. By examining its fundamental mechanisms and operational intricacies, we aim to illuminate the unique attributes and strengths of AGP in modeling and solving complex optimization problems like container port truck dispatching.

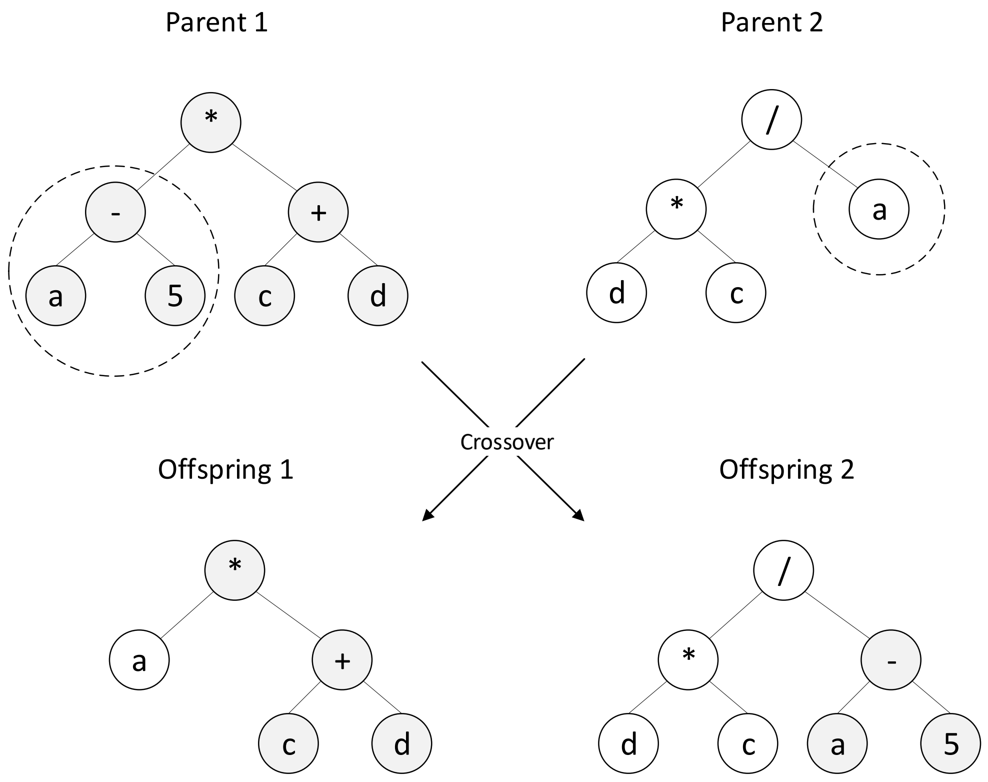


Figure 3.2: Crossover Operation of AGP & LGP

3.1.1 Crossover

The crossover operation takes two parental individuals selected through `tournament selection` and produces two offspring using a `single point crossover` operation. An example is illustrated in Fig. 3.2, where a subtree of parent 2 is combined with parent 1 to generate offspring 1, whereas offspring 2 has resulted from a merge of subtree 1 into parent 2.

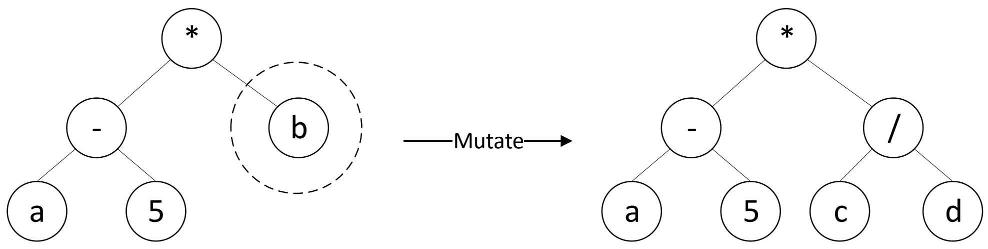


Figure 3.3: Mutation Operation of AGP & LGP

3.1.2 Mutation

The mutation operation takes one individual as input and generates a new offspring by a slight modification. A mutation point is randomly selected to grow a new randomly generated subtree that keeps the whole tree within the depth limitation, as illustrated in Fig. 3.3.

3.1.3 Depth Restriction

To avoid the bloating problem, researchers usually set a depth limit to the GP trees and discard or prune any result that exceeds the limits. We used two approaches in this study. The first method introduces a penalty term into the fitness function to penalize individuals who are too deep or have too many nodes. The second method sets a maximum depth of subtrees for crossover and mutation operations to generate resulting offspring within the required depth limit.

Incorporating principles inherent to natural selection, AGP initiates its evolutionary process with a randomly generated population. This population, representing diverse solutions, is rigorously assessed based on their fitness values. Each individual's fitness reflects its adaptability to the prevailing environment and its ability to solve the problem at hand.

In the context of the container port truck dispatching issue, the primary metric of fitness is denoted as TEU/h, which encapsulates the throughput efficiency of the port. Specifically, TEU/h quantifies the number of containers (Twenty-Foot Equivalent Units) that can be efficiently dispatched within an hour of port operations. A higher TEU/h indicates a more efficient truck dispatching strategy, and conversely, a lower value underscores potential inefficiencies. Therefore, this metric is a crucial determinant in gauging the efficacy of truck assignment strategies contrived by each individual in the population.

Adhering to genetic algorithm tenets, the premise of GP is rooted in the belief that proximity to a high-performing individual indicates the presence of more adaptive solutions. This mirrors the principle of natural selection, where individuals with superior adaptive traits tend to propagate these traits through generations. Drawing parallels, in the realm of AGP, individuals with higher fitness values (TEU/h) are construed to possess 'genes' or components of strategies that align better with the environmental demands.

By selectively breeding these elite individuals and applying evolutionary genetic operations such as crossover, mutation, and selection, the GP algorithm aims to refine the population iteratively. Over successive generations, the intent is to gravitate towards solutions that manifest enhanced adaptability and performance. This progression, grounded in the principles of evolution, ensures that AGP continually optimizes truck dispatching strategies, fostering diversity and optimization in its quest for the most adaptive solutions.

GP, being a form of hyper-heuristic, differentiates itself profoundly from conventional genetic algorithms (GAs). This distinction primarily lies in the solutions they generate during the evolutionary process. Instead of directly evolving the strategies for container port truck dispatching, GP concentrates on evolving methods or heuristics that can produce these strategies. In simpler terms, while GAs might emphasize generating a direct sequence of tasks for each truck, GPs are concerned with creating a programmatic approach to derive such task sequences.

This elevated approach of GP offers it several inherent advantages. It doesn't search within the solution space, as typical algorithms would. Instead, GP operates at a higher dimension, navigating the heuristic space. This means the evolved individuals (methods or heuristics) have enhanced robustness. Instead of being bound to a specific solution, they have the adaptability to generate various dispatching strategies contingent upon the prevailing environmental conditions.

The conventional static methods necessitate a renewed search in the solution space for every distinct task, rendering them potentially inefficient and less adaptable. On the other hand, the hyper-heuristic nature of GP allows it to be environmentally adaptive. By evolving methods that can generate solutions rather than the solutions themselves, GP ensures a quicker solution derivation time and broader applicability across different scenarios, thereby achieving enhanced adaptability and efficiency in diverse operational landscapes.

Meanwhile, the complex nature of the container port truck dispatching problem necessitates an approach that is not only versatile but also adaptive to the dynamic environment of ports. Conventional methods, while effective in certain scenarios, often lack the flexibility required to address the myriad of contingencies and evolving challenges that ports face daily (Kumari, 2021; Pluhacek et al., 2018).

GP emerges as a standout candidate in this context. Its inherent ability to evolve heuristics rather than concrete solutions, provides a more encompassing and adaptive framework. Instead of being confined to a set strategy, GP can generate diverse dispatching strategies based on the prevailing circumstances. This adaptability ensures that the solutions are not just effective for a single snapshot of the problem but are versatile across varying operational conditions.

Moreover, GP's focus on evolving methods that produce strategies, rather than the strategies themselves, enables a broader search space and greater adaptability. This not only reduces the computation time significantly but also ensures a higher degree of robustness in the face of dynamic challenges.

In essence, we employ GP for the container port truck dispatching problem because it promises adaptability, efficiency, and the breadth of its solution space. Its hyper-heuristic nature ensures that the generated solutions are optimized for the present and adaptable for the future, making it an indispensable tool in the ever-evolving landscape of container port operations.

3.2 Logic Genetic Programming

In the truck dispatching problem, various problem parameters/features have different degrees of influence in different scenarios. For example, the average quay crane load time and average quay crane unload time have more influence in scenarios dominated by loading and unloading operations, respectively. An AGP method would struggle to deal with such cases. This can be illustrated, as an example, by a piecewise-linear function in (3.1), where the value of x decides the scenarios and the corresponding results. To handle such problems, researchers usually choose the GP with logic operators (LGP).

$$y = \begin{cases} x & (x < 1) \\ x^2 & (1 \leq x < 3) \\ x^3 & (3 \leq x < 5) \\ x^4 & (5 \leq x < 7) \\ x^5 & (7 \leq x) \end{cases} \tag{3.1}$$

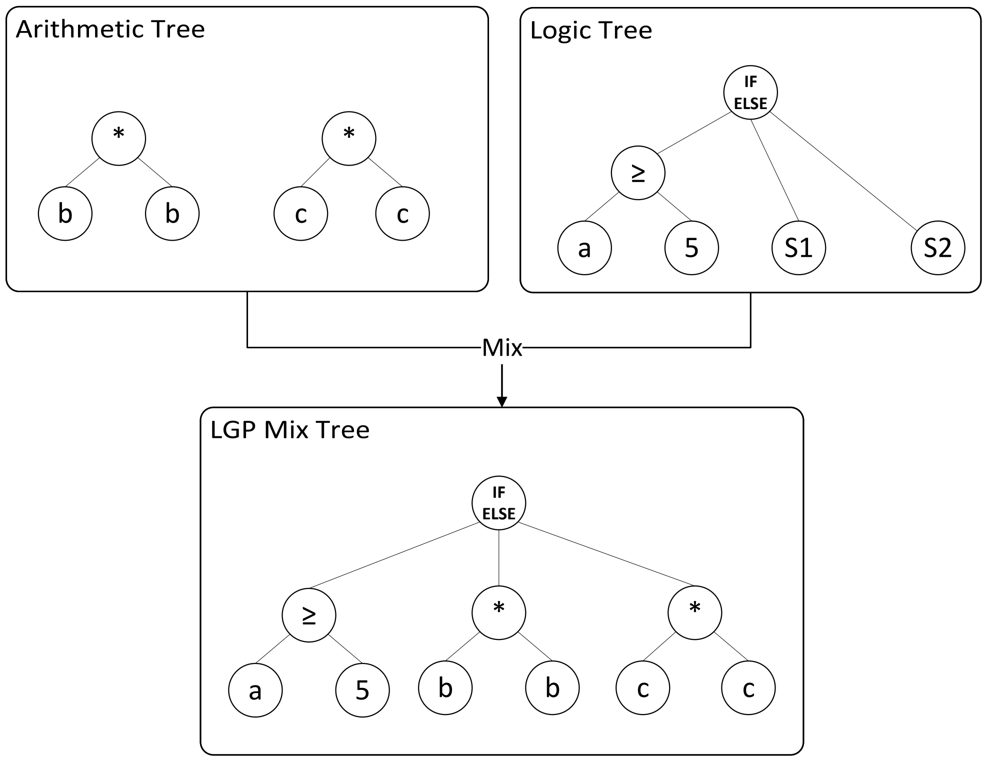


Figure 3.4: LGP structure

Although AGP can address the shortcomings of manually crafted heuristics by evolving parameterized heuristics from historical data, the resulting solution can be extremely complex and ineffective when handling problems involving multiple scenarios (i.e., when the distribution of random variables changes over time). For such problems, using some discrete utility functions is more elegant and efficient. LGP combining logic expressions with algorithmic trees presented in the previous section, different subtrees can be generated for different scenarios, which improves the performance of the algorithm and produces results that are adaptive to complex multi-scenario problems. This ability to adapt to different scenarios using a combination of the logic tree with arithmetic trees follows the general framework underlying hyper-heuristics (Burke et al., 2010a), and the approach is thus also named as a GPHH.

LGP trees with a positive probability of generating several logic trees at the top to select several arithmetic trees at the bottom (Fig. 3.4). LGP shares similar crossover and mutation operations with AGP but has additional operators designed for the logic tree, as shown in Table 3.1. Among these operators, the majority are binary, having two inputs and producing one output, such as "+, -, *, /, >=, <=, and, or, max, min". Exceptionally, the "IF_ELSE" operator has three inputs and selects either the second or third input as the output based on the value of the first input. Additionally, "max" and "min" are auxiliary operators used specifically within the context of real port problems.

Moreover, a loosely-typed GP is used in LGP, which allows arithmetic and logical operations to be combined freely. When performing logical calculations, numbers greater than 0 are treated as logically true, and vice versa as logically false. In this way, after the introduction of the "IF-ELSE" operator, different subtrees can be selected for calculation through the logical values of the previous decision tree. Therefore, some multi-scenario problems difficult to encode by a single depth-constrained arithmetic tree can now be more effectively addressed in LGP. For example, with the assistance of the "IF-ELSE" operator and comparison operators, we can evolve a simple LGP tree to represent the discontinuous function in (3.1) fairly easily.

In the multifaceted landscape of the truck dispatching problem, different parameters and features exert varied influences depending on the prevailing scenario. For instance, parameters like the average quay crane load time and its unload counterpart assume paramount importance in contexts chiefly characterized by loading and unloading actions, respectively. Traditional methods, such as AGP, often find themselves grappling ineffectively with these shifting weights and nuances. This dynamic can be elucidated by considering a piecewise-linear function showcased in (3.1). In this function, the parameter x serves as a determinant, dictating the

Table 3.1: AGP, LGP and CD-GPHH Operators

Name	**Label**	**Description**	**Algorithm**
add	+	Add operation	AGP, LGP, CD-GPHH
sub	-	Minus operation	AGP, LGP, CD-GPHH
multiply	*	Multiplication operation	AGP, LGP, CD-GPHH
divide	/	Division operation (protected)	AGP, LGP, CD-GPHH
greater or equal	>=	Greater than or equal to	LGP, CD-GPHH
less or equal	<=	Less than or equal to	LGP, CD-GPHH
IF ELSE	if_else	Conditional operator (if-else)	LGP
and	&	Logic AND	LGP, CD-GPHH (only in real-world problem)
or	|	Logic OR	LGP, CD-GPHH (only in real-world problem)
max	max	Return the maximum value	LGP, CD-GPHH (only in real-world problem)
min	min	Return the minimum value	LGP, CD-GPHH (only in real-world problem)

relevant scenario and its consequential outcomes.

In response to such intricacies, the academic and research community has predominantly leaned toward genetic programming equipped with logic operators, popularly known as LGP. This choice is rooted in LGP's inherent versatility and capacity to address scenario-specific challenges.

Though AGP brings commendable capabilities to the table, especially in redressing the limitations intrinsic to manual heuristics, by molding parameter-driven heuristics sourced from historical data, its solutions often become intricate mazes. These solutions, when confronted with problems that oscillate across multiple scenarios - especially where there's a temporal evolution of the distribution of random variables - can turn out to be convoluted and ineffectual. In these contexts, the elegance and efficiency of discrete utility functions shine through as a more compelling choice.

Delving deeper into the capabilities of LGP, its amalgamation of logical expressions with algorithmic trees (as delineated in preceding sections) allows it to spawn different subtrees attuned to distinct scenarios. This modular approach significantly bolsters the algorithm's efficacy, enabling it to churn solutions that seamlessly navigate the complexities inherent in multi-scenario challenges. Such dexterity in adaptation, born from the harmonious blend of logical and arithmetic trees, echoes the foundational principles of hyper-heuristics. This congruence with the hyper-heuristic paradigm, as documented by (Burke et al., 2010a), has bestowed upon this methodology the moniker of GPHH.

A closer inspection of LGP trees reveals their penchant for generating, with substantial probability, multiple overarching logic trees. These logic trees, positioned strategically at the apex, serve as gatekeepers, orchestrating the selection of several arithmetic trees nested below, a relationship vividly illustrated in Fig. 3.4. While the crossover and mutation mechanisms of AGP and LGP share common threads, LGP introduces an enhanced set of operators, meticulously tailored for its logic tree component, as cataloged in Table 3.1.

Furthermore, LGP leverages a loosely typed GP framework, facilitating the unconstrained melding of arithmetic and logical computations. Within this paradigm, numerical values exceeding the zero threshold are construed as logical truths, with their antitheses being designated as logical fallacies. This interpretative mechanism becomes especially pivotal upon incorporating constructs like the "IF-ELSE" operator. With this in play, different subtrees are marshaled into action based on the logical determinations of antecedent decision trees.

Consequently, challenges previously stymied a solitary depth-constrained arithmetic tree due to their multi-scenario intricacies can now find their nemesis in LGP. With tools such as the "IF-ELSE" operator and a battery of comparison operators, LGP can effortlessly evolve to epitomize the fragmented function encapsulated in (3.1) and the complex multi-scenario container truck dispatching problem.

Building upon our prior analysis, it's equally vital to appraise the container port truck dispatching problem through the lens of LGP's strengths. Although AGP offers specific advantages, several nuances and complexities inherent to the truck dispatching realm underscore why LGP might, in fact, be the more robust and adaptive solution.

- Multi-scenario Adaptability: Ports, especially global hubs, encounter many scenarios. From variations in ship sizes, and differing container types, to fluctuating cargo loads, the operational environment is riddled with dynamism. LGP's ability to seamlessly address multi-scenario problems equips it to craft dispatching strategies that can pivot and adapt according to varying operational landscapes.

- Logic Expressions with Algorithmic Trees: By combining logical operations with arithmetic constructs, LGP crafts a more versatile problem-solving framework. This hybrid approach is well-suited for the dispatching problem, allowing the system to make decisions based on numerical computations (like truck wait times or crane efficiency) and logical constraints (like operational rules or safety regulations).

- Depth and Breadth of Analysis: While AGP operates more on the surface level, LGP dives deeper, dissecting problems with greater granularity. In the context of truck dispatching, this means not just identifying which crane a truck should go to, but also assessing the sequence, considering the type of containers, predicting potential bottlenecks, and so forth.

- Loosely-typed GP: LGP's implementation, which allows arithmetic and logical operations to merge freely, provides a flexible decision-making canvas. For instance, in container port operations where decision parameters might not always be strictly binary (dispatch or don't dispatch) but can be influenced by many intermediate factors, LGP's loosely typed structure offers an edge.

- Robustness in Complex Environments: Ports can be volatile environments with sudden changes in weather, unexpected equipment malfunctions, or unforeseen logistical challenges. LGP's layered decision-making approach, utilizing logic trees atop arithmetic ones, ensures that the algorithm remains resilient to sudden changes, dynamically recalibrating strategies as scenarios evolve.

- Efficient Handling of Discontinuities: As highlighted previously, LGP's embrace of operators like "IF-ELSE" makes it adept at dealing with discontinuous functions or non-linear challenges. In the realm of truck dispatching, where operations are often non-linear and dependent on a multitude of interconnected factors, LGP's ability to seamlessly handle such discontinuities becomes invaluable.

- Reduced Need for Frequent Retraining: Given LGP's comprehensive decision-making framework, once trained, it might not require as frequent retraining as simpler models. It can generalize better across a broader spectrum of scenarios, ensuring consistent performance even as the operational environment of the port undergoes minor shifts.

In summation, while both AGP and LGP offer commendable tools to tackle the truck dispatching conundrum, LGP's multi-faceted approach, blending logic with arithmetic and offering depth of analysis, makes it particularly well-poised to handle the intricacies and dynamism of container port operations.

3.3 Cooperative Double-Layer Genetic Programming Hyper-Heuristic

In tackling the multifaceted and dynamic nature of seaport terminal truck scheduling, the Genetic Programming (GP) paradigm has exhibited potential. However, it's imperative to harness the strengths of logic operators within GP without succumbing to the perils of an exponentially growing search space. Taking cues from the existing literature on cooperative co-evolution GP (Nguyen et al., 2013; Potter and Jong, 2000) and innovative

GP representations (Nguyen et al., 2012; Hoai et al., 2006; Bi et al., 2020), this section unveils a novel methodology: the Cooperative Double-layer GP Hyper-heuristic (CD-GPHH). The hallmark of this approach is its bifurcation of scenario grouping and dispatch ordering into two distinct yet collaborative subpopulations. This structural division enhances the solution's interpretability and quality, particularly for intricate dynamic vehicle scheduling predicaments.

Under the CD-GPHH framework, GP individuals inhabit two collaborative echelons: the high-scenario layer and the normal-calculation layer. While both layers have distinct mutation and crossover strategies, they operate in synergy. The onus of the high-scenario layer individuals is to discern which counterparts from the normal-calculation layer should be activated to craft solutions for specific scenarios.

Despite LGP's prowess in managing intricate discontinuous functions and adjusting to multiple scenarios, its dependability remains a matter of concern. A noticeable portion of the LGP population struggles to construct multi-scenario adept solutions innately. The core challenge here is the substantial expansion of the search space due to the inclusion of logic operators, making it arduous for LGP to attain commendable convergence within a plausible computational timeframe. Our empirical analysis further accentuates this observation: LGP, while often producing noteworthy results, doesn't consistently align with the reliability prerequisites of industrial settings. Moreover, the inherent intertwining of logic and arithmetic constructs within an LGP tree hampers its clarity. This lack of lucidity became evident during our field tests at the Ningbo Port's real-world container terminal. The intricate LGP-derived solutions were met with skepticism by the port operators, who found them too convoluted for pragmatic applications.

In practical problems, performance metrics are typically separable. Take the truck dispatching problem for ports as an example, where operators usually consider different scheduling strategies based on, for example, task categories, yard conditions, the number of trucks available for dispatch, etc. In fact, these factors are high-level scenarios that are generally not directly involved in the formulation of task ranking rules; they instead play a role in the selection of different policies. Some researchers exploit these decision variables by grammar-based LGP to generate individuals with scenario distinguish ability (Nguyen et al., 2012; Sobania, 2021; Moyano and Ventura, 2022; Whigham et al., 1995). However, the grammar-based GP only adds grammar filtering to the normal LGP, not explicitly separating the scenario selection from the calculation layer. As a result, these factors do not always appear in the scenario layer to play a decisive role in scenario selection without presetting. Instead, they are often embedded in the calculation layer, which can cause unreliable performance.

Consequently, to reduce the size of the search space and to improve performance, it is necessary to separate scenario information from dispatch rules, leading to our proposed CD-GPHH method, which reduces the search space size by separating arithmetic trees and scenario trees into two different layers. The concept underlying CD-GPHH is to evolve the scenario by grouping trees and truck dispatch trees concurrently but at two different layers. More specifically, each individual in CD-GPHH has a scenario layer and a calculation layer (Fig. 3.5). The scenario layer contains logic trees for scenario clustering purposes, and the calculation layer includes arithmetic trees that share structures similar to those in the AGP method. Each logic tree in the scenario layer is bound to an arithmetic tree from the calculation layer and grouped as a rule. In the truck dispatching problem for ports, when a tree in the scenario layer evaluates to `true` (greater than zero), then the corresponding tree in the calculation layer is invoked to compute utility scores for different truck–task assignments. Notably, thanks to this layered structure, our CD-GPHH is also more comprehensible while enhancing the quality of the resulting solutions.

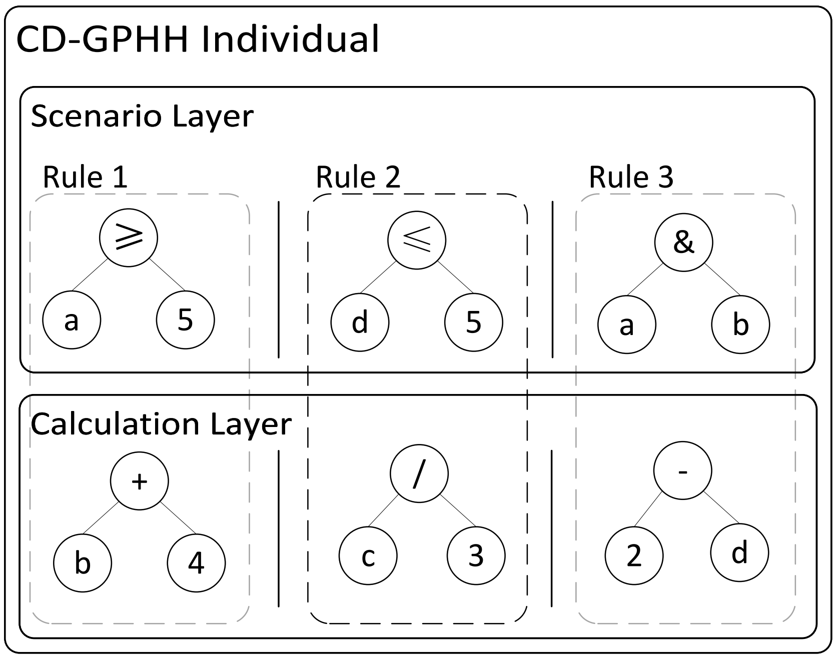


Figure 3.5: CD-GPHH individual structure

Note that, in CD-GPHH, trees in both the scenario and calculation layers are bound into `rules` for easier computation during implementation. The algorithm operates on a specific *rule* before processing the two trees inside each rule. Moreover, since the number of scenarios (number of rules) was introduced as a hyper-parameter in CD-GPHH, we extend the traditional mutation operation in GP to enable it to learn and adjust automatically. These three operations are detailed as follows:

3.3.1 Crossover

The crossover operation takes two individuals (parents) as inputs. As illustrated in Fig. 3.6, one random rule of each parent is then selected, and

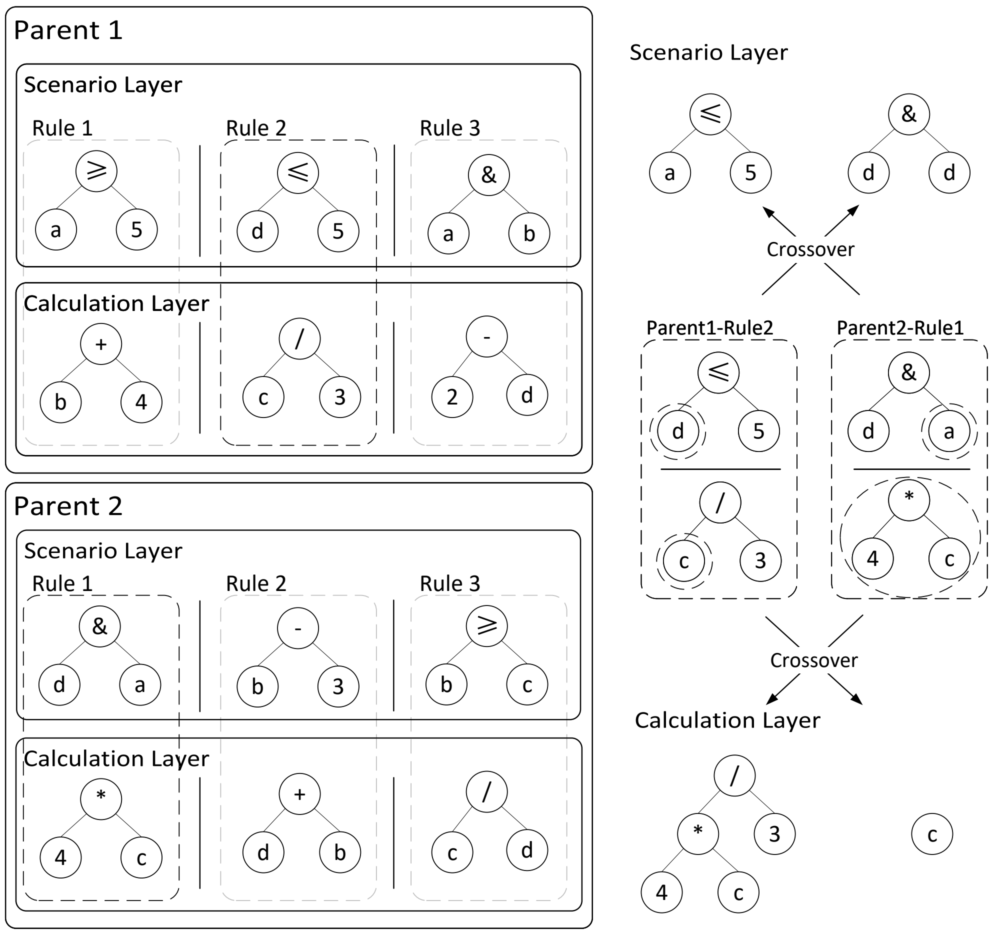


Figure 3.6: Crossover Operation of CD-GPHH

single-point crossover operations are applied to both layers independently, resulting in two logic trees and two arithmetic trees that can form four different rules. When the four rules are inserted back into the two parents to replace the previously selected rules, eight new offspring are created. However, to maintain the diversity of the population and prevent the same pair of parents from producing too many offspring, two randomly selected offspring are retained in the next generation.

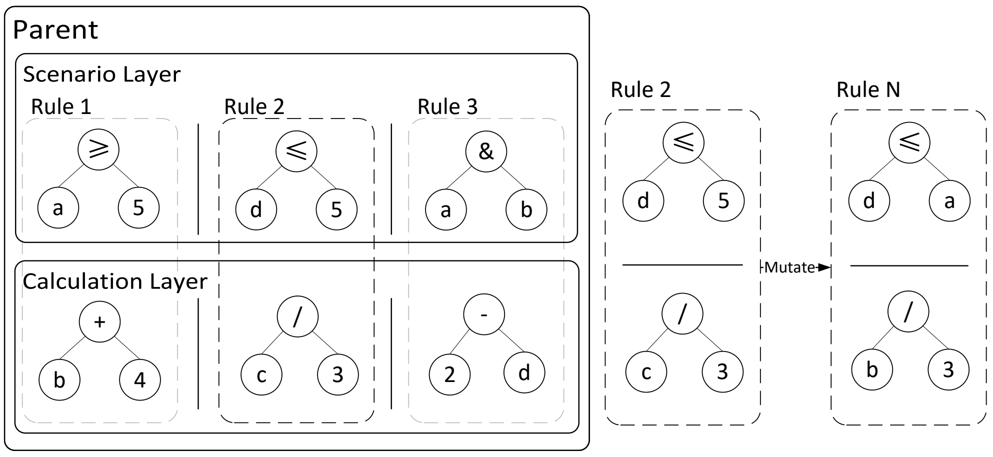


Figure 3.7: Mutate Operation of CD-GPHH

3.3.2 Mutation

The standard mutation in CD-GPHH is applied to one parent selected by the tournament. A rule in the parent is randomly selected for mutation. For example, in Fig. 3.7 is rule 2 selected for mutation. Either the scenario layer tree or the calculation layer tree, or both are modified to form a new rule (same as AGP). This new rule is then inserted into a random location of the parent to generate a new individual (hence, this new individual has one more new rule than the parent). Meanwhile, to maintain diversity and dynamically adjust the number of rules, between zero to two randomly chosen rules in this new individual are subject to removal. Finally, if the total number of rules in this new individual exceeds the rule count limit, another randomly selected rule will be removed.

3.3.3 Solution Decoding in CD-GPHH

In CD-GPHH, decoding a solution involves testing all logic sub-trees (except the last) in the scenario layer in sequence until a sub-tree evaluates to `true`. The corresponding calculation tree will be chosen to estimate the performance of different task assignments. If no logic tree evaluates to `true`, without evaluating the last logic tree, the last (default) calculation tree will be used.

3.4 Experiments and Results Analysis

We evaluated the performance of our proposed CD-GPHH against a conventional AGP as well as an LGP for solving multiscenario problems. We first compared their performance for the simple multi-scenario function fitting problem described in Section I. Subsequently, we conducted extensive experiments for the real-life truck dispatching problem for marine container terminals. For the truck dispatching problem, we also compared our CD-GPHH against the traditional heuristic method used in real-world and a manually crafted heuristic reported in (Chen et al., 2016). Since parameter tuning is not the key focus of this paper, we just adapt the common settings for all three GP-based algorithms, as listed in Table 3.2. To ensure fairness in comparison, the depth limitation for AGP and LGP is set to 10, whereas for CD-GPHH, it is reduced to 5. Given the new structure of CD-GPHH incorporates multiple internal rules, the rule amount within CD-GPHH is set as 1 to 10.

Table 3.2: GP Initialisation Parameters

Population Size	1024
Max Generation	300
Crossover Rate	0.6
Mutation Rate	0.3
Reproduction Rate	0.1
Tree Initialization Method	Ramped half-and-half
Selection Method	Tournament Size 7
Tree Depth Restriction	10 (AGP&LGP) / 5 (CD-GPHH)
Long Individual Penalize Scale	0.0001
Rule Amount Restriction (CD-GPHH only)	1-10

Table 3.3: AGP, LGP, and CD-GPHH Test Results on Problem (3.1)

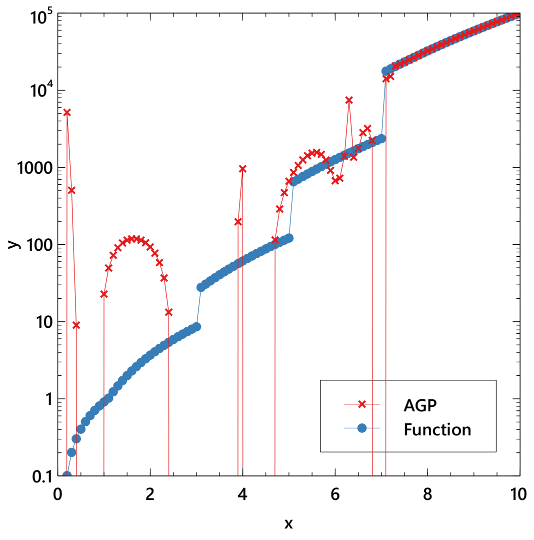
		AGP	**LGP**	**CD-GPHH**
Standard Deviation	**Min**	**288.46**	**4.68**	$\mathbf{2.02 \times 10^{-12}}$
	Mean	447.21	248.24	0.042
	Max	680.65	940.08	0.22

3.4.1 Simple Multi-scenario Function Fitting Problem

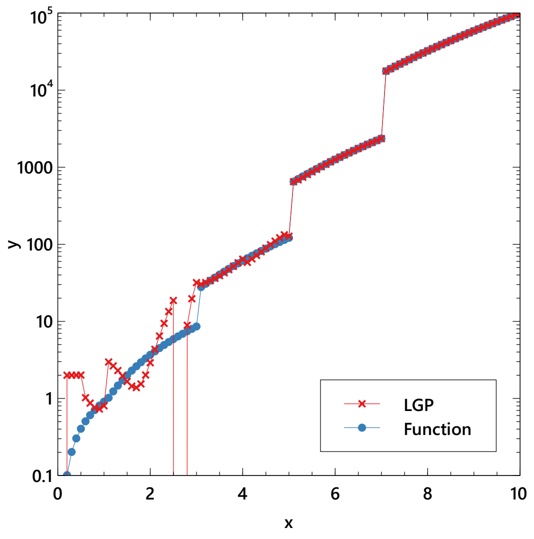
In this experiment, the task is to fit function (3.1). There is one input variable x and one constant that corresponds to two terminals in all GP methods. The variable terminals were set to an actual value during the calculation process, whereas a constant integer between 0 and 10 was set as the terminal. During each test, 100 randomly pre-generated instances of different values of x were used as the test set. Fitness was defined as the standard variance between the fitted function and the original function. In other words, a fitness closer to zero indicates a good solution.

The function (3.1) is utilized in this study as a benchmark for a straightforward function fitting problem, allowing for comparing various GP algorithms' performance in addressing multi-scenario challenges. Although function (3.1) features a single input and output, it is designed to represent multiple distinct scenarios based on varying values of x. This setup closely mirrors the real-world environment of port truck dispatching. Different conditions—weather, time, the number of trucks and ships, and congestion at intersections—necessitate tailored heuristics for dispatching to maximize port operational efficiency.

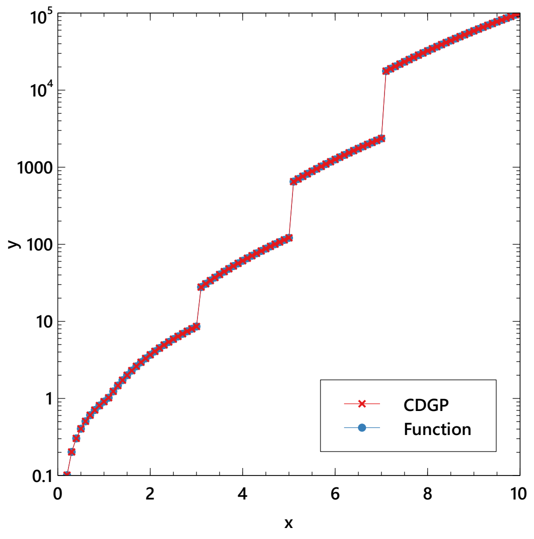
Moreover, while the problem outlined in function (3.1) might seem simplistic, it encompasses the essence of multi-scenario and multi-environment outputs. Its limited number of variables allows the results to be effectively visualized and understood through graphical representation. Hence, we chose this equation as a straightforward example to compare the performance and interpretability of AGP, LGP, and the CD-GPHH introduced in this Chapter.



(a) AGP



(b) LGP



(c) CD-GPHH

Figure 3.8: AGP, LGP, and CD-GPHH best result in fitting Eq. (3.1) compared with the original function.

Table 3.3 presents the statistical results from all three GP-based methods in 30 tests. CD-GPHH performed significantly better than AGP and LGP. As shown in Fig. 3.8, comparing the best results against the original function, AGP performs the worst, fitting only the case where x is greater than 7, while LGP fits the range from 3 to 10. When x is less than 3, the fitting errors lead to a relatively small variance in y (equivalent to a small variance in individuals' fitnesses). LGP, therefore, does not fit well in the range of 0 to 3 region in Fig. 3.8, while CD-GPHH has replicated almost 100% of the original function, indicating the potential benefits of a predefined hierarchical structure for multi-scenario problems. The simplified best-performing individuals of each method are as follows:

- AGP:
  $(3920805*x^{10}-90391140*x^{9}+613811750*x^{8}+zoo*x^{8}+563298482*x^{7}-22692826327*x^{6}+78451667966*x^{5}-16277641800*x^{4}+-218951495280*x^{3}+62156445120*x^{2}+14180443200*x)/(3991680*x^{5}-103783680*x^{4})+1010892960*x^{3}-4372885440*x^{2}+7090221600*x$

- LGP:
  $if\_else(-(5>=2*x+(x<=7))/5+(7<=x)/5, x*(x^{4}-1)+x, x*((x-if\_else(if\_else(if\_else((x-3)/(2*x),(5-x<=-(5>=2*x+(x<=7))/5+(7>=x)/5),1-if\_else((x^{2}*(x<=8)+(x<=0)),0,1))+1, if\_else((3<=2*x),x-11/10,1),0),1,7)+5)*(40*x^{2}-x^{3}+40*(9>=x)-440)+320)/40)$

- CD-GPHH:

Rule	Scenario	Calculation
1	$7 <= x$	x^5
2	$-16 * x^2 * (x-10)^2 + 51 * x - 6)/(4 * x * (x-10)) >= (x + 3/x >= 4)$	x^2
3	$x^2 + x <= 4$	x
4	$x - 10/x <= 3$	x^3
5	$x > 3/4$	x^4

Obviously, the results by AGP and LGP are hard to interpret and differ greatly from the true function in (3.1). Although LGP evolved the "IF-ELSE" and relational operators that were in the original expression, the results are rather confusing and do not fit the original function well. By contrast, with the help of its bi-level structure, CD-GPHH obtained a concise and easy-to-understand function that is almost identical to the original one.

Among the best evolution results of the three methods in Fig. 4.4, CD-GPHH has escaped the local optimum trap and converged to the global optimum at the 150th generation, while both AGP and LGP ended up with a poor result. In general, under the evolutionary framework designed in this research, CD-GPHH takes advantage of two cooperative populations and obtains well-performed, concise, and understandable results, confirming its superiority in solving the simple multi-scenario function fitting problem.

3.4.2 Results for Real-life Truck Dispatching

The three different GP methods are further evaluated on the truck dispatching problem described in Section 1.2.2. The 14 features in the proposed

Table 3.4: Features of AGP, LGP, and CD-GPHH in Real-World Truck Dispatching Problem

Name	**Acronym**	**Description**
travel_time	tt	Travel time form truck to task start crane
operate_type	ot	The ship load/unload task type (0 for load and 1 for unload)
dispatch_type	dt	The task dispatch type (0 for normal tasks and 1 for tasks need to be merged)
yard_crane_type	yct	The yard crane type (0 for normal crane and 1 for remote-controlled crane)
total_truck_num	ttn	Total trucks number can be dispatched
num_to_min_truck	ntmt	The difference between the quay crane trucks number and the minimum trucks number
start_node_truck_num	sntn	Total trucks number of the task start crane
end_node_truck_num	entn	Total trucks number of the task end crane
start_node_wait_truck_num	snwtn	Waiting trucks number of the task start crane
end_node_wait_truck_num	enwtn	Waiting trucks number of the task end crane
remain_task_num	rtn	Remaining tasks number of the task related quay crane
average_load_time	alt	Average load time of the crane
average_unload_time	aut	Average unload time of the crane
Constant Number	-	Random constant number

Table 3.5: Traditional Heuristic Methods for Truck Dispatching Problems

Name	**Acronym**	**Description**
fixed	-	Dispatchings are according to the binding QCs
manual	-	Select the task though manual crafted heuristic (Currently used in Meishan Port, Ningbo)
random	-	random dispatch
first-in-first-out	FIFO	Dispatchings are sequenced first-in-first-out
shortest traveling time	STT	Select the task with the shortest traveling time
longest traveling time	LTT	Select the task with the longest traveling time
most task remaining	MTR	Select the task with most task remaining
smallest task remaining	STR	Select the task with smallest task remaining

CD-GPHH are shown in Table 3.4. The other settings are given in Table 3.2. To better tackle this complex real-world problem, the four new operators in Table 3.1 were added. The performance of the proposed three GP methods is compared with other traditional heuristic methods (Table 3.5) in this subsection.

To update the environment state values of these features and evaluate the fitness of each individual in all GPs, we built an event-based simulator based on the mathematical model described in Section 1.2.2. The simulator interacts with the dynamic truck dispatching system (Fig. 1.2) to provide functions for evaluating the fitness of each individual and generating new environment data after each truck dispatch. It can simulate all events occurring in real-world truck dispatching in marine container terminals, such as vehicle movement, container loading and unloading, and real-time

Table 3.6: AGP, LGP, and CD-GPHH Training Results (units/h)

	Fixed	**Manual**	**Random**	**FIFO**	**STT**	**LTT**	**MTR**	**STR**	**AGP**	**LGP**	**CD-GPHH**
Set1	106.05	106.87	100.01	94.58	71.39	91.88	108.75	54.89	124.03	126.09	132.88
Set2	113.63	118.91	108.66	96.99	69.23	71.35	115.66	58.67	127.22	132.32	138.45
Set3	110.10	114.27	106.39	100.55	75.50	76.86	120.58	56.33	129.38	128.86	138.73
Set4	115.33	121.48	112.75	97.60	69.22	87.17	121.91	59.07	135.53	138.37	143.99
Set5	96.76	116.88	102.95	95.13	66.40	65.76	114.46	55.97	121.34	125.07	132.44
Mean	108.37	115.68	106.15	96.97	70.35	78.60	116.27	56.99	127.50	130.14	137.30
Imp.	0.00%	6.74%	-2.05%	-10.52%	-35.09%	-27.47%	7.29%	-47.42%	17.65%	20.09%	26.69%
	-6.74%	0.00%	-8.98%	-19.30%	-64.44%	-47.17%	0.51%	-102.99%	9.27%	11.11%	15.75%

Table 3.7: AGP, LGP, and CD-GPHH Test Results (units/h)

	Fixed	**Manual**	**Random**	**FIFO**	**STT**	**LTT**	**MTR**	**STR**	**AGP**	**LGP**	**CD-GPHH**
Set6	102.50	105.35	107.16	88.14	66.96	75.89	106.74	55.27	117.86	123.96	129.67
Set7	116.00	126.85	107.46	100.80	74.72	87.88	119.28	60.57	130.43	136.60	141.01
Set8	93.69	99.88	97.04	87.53	69.89	75.03	102.28	51.74	106.92	111.42	118.34
Set9	98.45	106.01	96.96	96.57	68.81	85.55	108.81	53.89	112.58	121.31	125.31
Set10	112.12	121.37	106.84	101.59	76.23	85.39	121.85	57.07	121.95	125.76	133.78
Mean	104.55	111.89	103.09	94.93	71.32	81.95	111.79	55.71	117.95	123.81	129.62
Imp.	0.00%	7.02%	-1.39%	-9.20%	-31.78%	-21.62%	6.92%	-46.72%	12.82%	18.42%	23.98%
	-7.02%	0.00%	-8.53%	-17.87%	-56.89%	-36.54%	-0.09%	-100.85%	5.14%	9.63%	13.68%

data exchange with the dynamic dispatching system.

The data sets used in this experiment were extracted from the actual historical operational data at the Ningbo Meishan Port. The data sets simulate a typical situation where one container ship berths at the port to load and unload containers, and the port needs to complete the work of the ship as soon as possible to release the ship from the port earlier. Several instances were generated based on this situation, and each with one ship berth with six QCs. The number of trucks is set to be the actual number of trucks working during the data extraction time period, between 24 and 48, and the traffic map is shown in Fig. 1.1.

As aforementioned in Section 1.2.2, due to strict traffic regulations (mainly single-direction road segments), there are very few route options between QCs and YCs. Therefore, the truck travel time is pre-computed through the shortest path algorithm on the port road network, assuming an average truck speed of 8km/h. Meanwhile, the operating (load/unload) time of the crane for the container on the truck is uncertain.

We extracted 10 sets of historical task data from different time periods (ports have different operating scenarios at different times), with five sets for training (sets 1-5) and five sets for testing (sets 5-10). Each set (both training and testing) contains 10 instances in the same time period, with 200 tasks consisting of a mixture of both loading and unloading operations. For each training instance, 30 independent runs with different random seeds were conducted, leading to a total of 10 * 30 = 300 runs on each training set. The average results over these 300 runs for each set are presented in Table 3.6.

Recall that the objective is to maximize the number of tasks handled per unit time (hour). Therefore, larger values indicate better performance. The best-performing individuals from the three GP algorithms are evaluated on test sets 6-10. Each instance in the test sets was run only once (because the evolved GP trees are deterministic). However, since each set contains 10 instances, there are a total of 10*5 test instances that are not seen during GP training. Therefore, they represent significant robustness tests for all the methods. Table 3.7 provides the average results of all the methods across 10 test instances in each set as well as the averages across all 5 sets.

Below we include the best-performing individuals of AGP, LGP, and CD-GPHH. Note that these trees have been simplified for ease of interpretation.

- AGP:
  $(((entn + sntn) - rtn) + snwtn) + (ttn/((((((ttn + ot)/(((tt/yct) * rtn))) + ((rtn * alt)/(ntmt + alt))) - (((rtn + ntmt)/(entn + snwtn)) + ((yct + ttn) * (enwtn - sntn))))/((((entn + entn) * (snwtn * 2))/((aut/ttn) /(dt * sntn)))/(((ntmt + dt) - (tt * rtn)) + ((snwtn * aut) - (tt - sntn))))) + rtn))$
- LGP:
  $((((max(((ot|7|((entn/10)+ot))|min(max(2, rtn), -ntmt)), snwtn) <= ot)/10) * (entn * dt)) <= if\_else(rtn, tt, sntn)) * ((ttn/(((1 + if\_else (yct, 6, rtn)) * ((((sntn/max(dt, ntmt)) * (rtn \& snwtn))/snwtn) \& (sntn * dt))) + (((ttn/rtn)/((entn/if\_else(tt, sntn, enwtn)) \& enwtn)) >= ((10 - tt) <= (4 * entn)))))/(enwtn >= ntmt))$
- CD-GPHH:

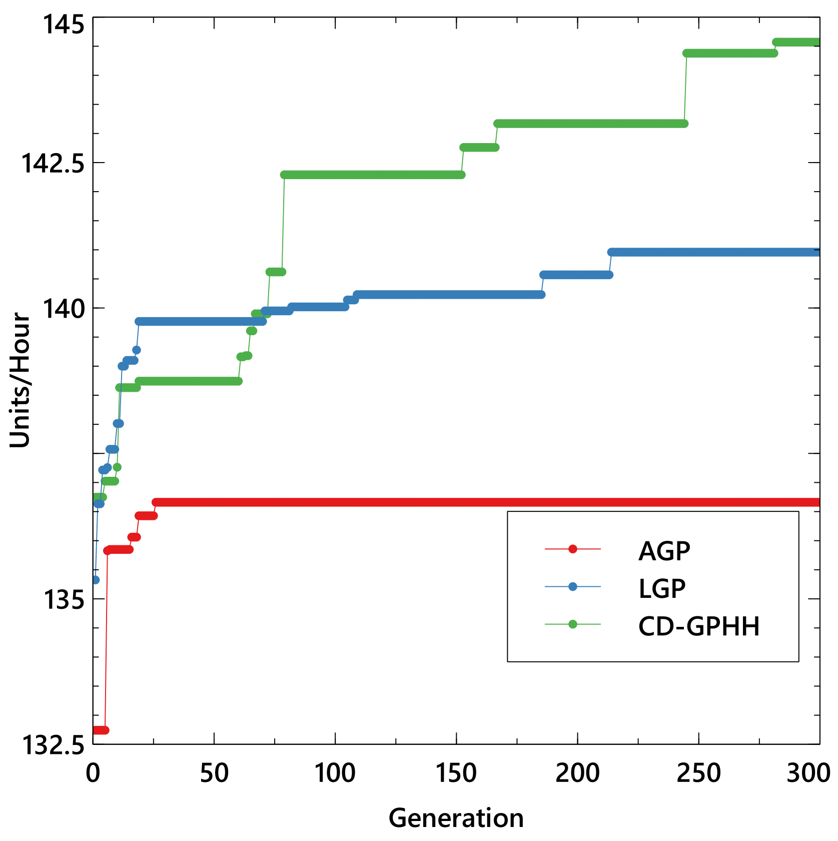


Figure 3.9: The performance of AGP, LGP and CD-GPHH in Training Set 4.

Rule	Scenario	Calculation
1	$((((ntmt >= sntn) - ntmt + entn)) + (ot + snwtn + min(ctn, tt)))\|((min(tt, enwtn) + max(dt, ot))/tt)$	$(dt >= enwtn) + max(entn, ctn) + rtn$
2	$ctn >= 5$	$snwtn/3 + alt * aut$
3	$(min(tt, ctn)/(ot\&snwtn) <= (ot + tt))/((dt >= entn) <= (ot <= sntn))$	$min(((enwtn <= entn) + ntmt), ((ctnentn) * ntmt))$
4	$max((entn >= 5), ntmt)$	$ctn * alt$

A t-test on the experiment ($\alpha = 0.001, p = 0.00$) demonstrates the superior performance of all three GP algorithms than the traditional heuristic algorithms, and CD-GPHH achieved the best. Using manual and fixed heuristic algorithms as benchmarks, the improvement percentages (Imp. Pct.) of AGP, LGP, and CD-GPHH are about 7%/15%, 11%/19% and 14%/25% , respectively. By further comparing the listed individuals of three GP algorithms, we can find that CD-GPHH not only produced more efficient heuristics but is also much more readable thanks to its double-layer structure. In contrast, AGP, and particularly LGP produced heuristics difficult to understand. The operator must further modify these heuristics in practice, while the heuristics produced by CD-GPHH have better usability.

In order to observe the evolution process of the three GP algorithms, one result of example training set 4 is plotted in Fig. 3.9. It can be seen that for the real-world multi-scenario problem, the performance of AGP without scenario grouping is quite limited, while LGP and CD-GPHH can achieve better results. In particular, CD-GPHH did not suffer from the obvious limitations of AGP and performed best in the end.

3.4.3 Truck Dispatching Under Special Scenarios

Although problem instances based on real-life data are important to evaluate the practicality of the proposed method, they are less useful for gaining useful insights due to the real-life complexities and the combinatory effect of several uncontrolled factors. In this subsection, we evaluate the performance of different methods under three different scenarios created artificially. In a container terminal, the multiple scenarios are associated mainly with the following dynamically changing factors.

- The operation times of the load and unload QC tasks along the berth line are practically known to follow different distributions. The unloading tasks are often less likely to be disrupted by truck delays due to the less strict precedence requirements. On the other hand, loading tasks must follow the predefined sequences exactly, and hence delays can propagate exponentially, causing significant QC waiting. Therefore, different dispatch policies are required for scenarios with tasks dominated by either load or unload tasks.

- Distribution of operation nodes at yard cranes (YCs) for the tasks is also crucial. Clustered operation nodes at a few YCs are more likely to lead to conflicts, as YCs need to support multiple QCs at the time. Specific policies are required to resolve these conflicts.

- The available number of trucks for dispatch is also important. When enough trucks are available, the priority should focus on reducing empty truck travel distances; otherwise, the priority is to avoid costly QC waiting.

Following these considerations, we use *operation_type*, *yard_crane_type*, and *total_truck_num* to distinguish these scenarios respectively. This is also based on the observation that over 75% of the of best-performing CD-GPHH individuals use all three features in the scenario selection layer. Therefore, we created 3 new data sets (sets 11-13), each containing 20 instances with 2 special scenarios based on these 3 scenario features. Individuals were trained (30 runs per instance) with corresponding scenario features and then to simulate the situations with or without scenario information.

The statistics in Table 3.8 demonstrate the significantly enhanced performance from CD-GPHH in the special scenarios data sets. There is no obvious impact on the performance of AGP without the scenario features.

However, the performance of LGP dropped by 2.9% when scenario features were excluded, compared with 8.1% from CD-GPHH. This justifies the effectiveness and importance of the scenario identification used in CD-GPHH on multi-scenario problems.

Table 3.8: AGP, LGP, and CD-GPHH Result in Special Scenarios Truck Dispatching Problem (units/h)

Set	**Feature**	**Manual**	**AGP**	**LGP**	**CD-GPHH**
Set11	With	107.46	113.85	122.64	131.69
	Without	107.46	114.44	118.32	119.68
Set12	With	105.02	107.67	115.51	125.72
	Without	105.02	105.17	113.33	113.71
Set13	With	99.38	104.96	110.69	123.34
	Without	99.38	106.00	106.28	112.74
Mean	With	103.95	108.83	116.28	126.92
	Without	103.95	108.54	112.64	115.38
Imp.	With	0.00 %	4.48 %	10.60 %	18.09 %
	Without	0.00 %	4.23 %	7.72 %	9.90 %

Finally, it was confirmed that CD-GPHH could produce better results than AGP and LGP in real-life multi-scenario problems. The generated results can also be understood and then modified by operators. According to our statistics, the average test time and training time of each generation of AGP, LGP, and CD-GPHH for 100 tasks is 0.02s / 0.7s, 0.02s / 0.73s, and 0.021s / 0.78s, respectively. This proved the improved performance of CD-GPHH does not require a significantly increased computational cost based on AGP and LGP. Although CD-GPHH has not yet been adopted in a real-life port, our manually crafted heuristic algorithm has been practiced at the Ningbo Port for years. Statistical analysis conducted by the port showed that the work efficiency increased by 8.1% and ship docking time decreased by 2.2%. This well-performed algorithm saved time, allowed for the operation of more ships, and in turn, increased the profit of the port company significantly. Our next plan is to work with the collaborators to fully evaluate and deploy the proposed algorithm in the real world.

3.5 Summary

This chapter embarked on an expedition to delve into the challenging domain of truck dispatching at marine container terminals, which presents itself as a crucible for innovative solutions due to its inherent complexities. Beginning with the AGP, which provided an initial groundwork for handling the container port truck dispatching problem, we quickly realized its limitations in the face of scenarios dominated by varying problem parameters and transitions.

The LGP, with its incorporation of logic operators, presented an advancement by exhibiting prowess in managing intricate discontinuous functions and multi-scenario problems. However, despite its enhancements over AGP, LGP lacked consistent reliability, especially in industrial environments characterized by real-world uncertainties.

It was these insights and challenges that led to the inception of the novel CD-GPHH algorithm. Distinguished by its cooperative double-layer structure, CD-GPHH champions a pragmatic approach in the real-life domain of marine terminals. With an architecture that embraces scenario transitions' dynamism, CD-GPHH efficiently exploits logic and arithmetic operators without succumbing to an overwhelming search space. This was evident in its superior performance, outclassing both AGP and LGP by an impressive margin of 8-10%. Furthermore, the clear demarcation between the logic and arithmetic layers in CD-GPHH enhanced the usability and legibility of the evolved heuristics.

GP, as a hyper-heuristic method, manifested strong generalizability by operating at a hyper-level, thus affording us a broad view of solution spaces. CD-GPHH, building upon the foundations of LGP, capitalizes on this strength of GP, offering an effective tool against multi-scenario challenges prevalent in truck dispatching tasks.

However, the journey of refining CD-GPHH is by no means at its terminus. Future research avenues beckon, especially in addressing its existing weaknesses like generalization issues and the existence of redundant subtrees. A confluence of human intuition and algorithmic rigor might provide the necessary impetus, guiding evolution more effectively. Additionally, tailoring redundancy removal algorithms specifically for CD-GPHH could pave the way for more streamlined and efficient solutions.

In conclusion, the marine container terminal truck dispatching domain, while fraught with challenges, presents fertile ground for innovation. The AGP, LGP, and CD-GPHH serve as testamentary milestones in this evolving journey, with each bringing its strengths and lessons. As we navigate forward, the fusion of human expertise with algorithmic advances holds the promise of even more refined and robust solutions for this intricate problem.

Chapter 4

Reinforcement Learning Assisted Method in Dispatching Strategy Generating

After we tested the manual heuristic, we found that this traditional expert heuristic consumes a significant amount of time and labor to build and adjust and can still not cope with diverse scenarios. Thus, auto-generated heuristics based on real-life data were proposed to adapt to complex scenarios in real-world operating environments. Genetic programming (GP) and reinforcement learning (RL) are considered capable of automatically learning parameter settings and dispatching methods for different scenarios based on historical data, thus achieving better scenario adaptability. However, it is disappointing to observe the poor performance of these two algorithms when directly applied to the dynamic container port truck dispatching problem Chen et al. (2023).

Consequently, we pivoted to a hyper-heuristics framework, distinguished by its enhanced robustness and generalization capabilities Burke et al. (2013). Based on the proven efficacy of deep reinforcement learning-based hyper-heuristics (Deep Reinforcement Learning-based Hyper-heuristics (DRL-HH)) Zhang et al. (2022) and genetic programming hyper-heuristics (GPHH) Chen et al. (2020) in simplistic training environments, we aim to extrapolate this effectiveness to unseen, more complex real-world testing environments. Considering the port's need for interpretability of dispatch methods, we introduced a framework that employs high-level heuristics to select explainable low-level GP heuristics training in different environments. This approach aggregates the intelligence of multiple GP heuristics, thereby dramatically improving the performance of DRL-HH and GPHH in complex real-world environments while eliminating the dependence on expert-designed heuristics.

In the realm of high-level heuristics in this framework, both GP and RL emerge as strong contenders due to their adaptability and data-driven capabilities. However, our experiments demonstrated that using GP as a high-level heuristic to select low-level GP heuristics (CD-GPHH) Chen et al. (2022) yielded promising results in small, generated datasets but faltered with more extensive, real-world datasets. To overcome this limitation, we propose a reinforcement learning-assisted genetic programming hyper-heuristic (Deep Reinforcement Learning-assisted Genetic Programming Hyper-heuristic (DRL-GPHH)). This new approach incorporates a set of GP-generated low-level heuristics and employs a reinforcement learning agent to act as the selector among these heuristics for various scenarios. Comparative evaluations indicate that DRL-GPHH outperforms its counterparts in simulated real-world conditions.

Nonetheless, it is noteworthy that the performance enhancement observed in DRL-GPHH was not remarkably superior to the results achieved by the best GP individual. This is because in DRL-GPHH, multiple GP evolved heuristics participate in the process, but during each decision point, only one GP individual is chosen to calculate the dispatching preference. Such an approach wastes the knowledge embedded in other unselected GP individuals, weakening the algorithm's performance and generalization capabilities. With this hypothesis, we propose a reinforcement learning-assisted genetic programming ensemble hyper-heuristics (Deep Reinforcement Learning-assisted Genetic Programming Ensemble Hyper-heuristics (DRL-GPEHH)) method, which uses a reinforcement learning agent to assign different weights to different GP individuals during each dispatch and subsequently combines the results of all GP heuristics according to their weights to produce the final dispatching solution.

By seamlessly integrating the preference discrimination power of multiple auto-evolved GP heuristics, DRL-GPEHH excels in complex dynamic optimization tasks, exemplified by port truck dispatching. This framework not only performs outstandingly during training but also remains resilient against unseen test data, highlighting its potential for addressing various real-world dynamic optimization challenges. DRL-GPHH and DRL-GPEHH, distinctive from traditional DRL, DRL-HH, or GPHH, mark a pivotal leap in generative, fully automated optimization methods. Our proposed method significantly contributes to solving real-life optimization problems in the presence of uncertainties and dynamics.

The primary content of this chapter is:

- We propose DRL-GPHH, an approach that employs DRL to select GP-generated low-level heuristics. This approach effectively increases the adaptability of the algorithm, eliminating the dependence on expertly designed low-level heuristics inherent in DRL-HH, resulting in a much higher level of automation in algorithm design.

- We further propose a novel GP ensemble hyper-heuristics framework DRL-GPEHH, which leverages the knowledge of multiple auto-evolved GP heuristics simultaneously during scheduling, leading to significantly improved performance and robustness compared to DRL-HH and DRL-GPHH.

- We design a multi-action proximal policy optimization (PPO) agent with a novel reward to effectively adjust the weights of several lower-level heuristics within an action, promoting collaboration and performance improvement.

- We conduct comprehensive experiments and ablation studies on benchmark instances, covering diverse scenarios and terminal configurations, to fully test the performance of DRL-GPEHH in comparison with its main variants, DRL-HH, DRL-GPHH, and a deep reinforcement learning-assisted ensemble hyper-heuristics (Deep Reinforcement Learning-assisted Ensemble Hyper-heuristics (DRL-EHH)). The results demonstrate the superiority of DRL-GPEHH in the dynamic truck dispatching problem.

- We analyze the differences between manual heuristic and GP-generated heuristic ensemble methods and explain why a continuous weight adjustment is necessary for GP ensembles to achieve higher performance, further highlighting the compatibility of DRL-GPEHH with GPHH integration and establishing a novel approach for RL and GP cooperation.

The rest of this chapter is organized as follows. The proposed DRL-GPHH and DRL-GPEHH methods are delineated in Section 4.1 and 4.2. Section

4.3 delineates the experimental outcomes and provides ablation & sensitivity analysis of DRL-GPEHH. Finally, conclusions are drawn in Section 4.4.

4.1 DRL-GPHH & DRL-HH

In recent years, Reinforcement Learning (RL) methods, particularly Deep Reinforcement Learning (DRL), have gained considerable traction in the scientific community due to their ability to successfully address a broad spectrum of complex optimization problems. They employ a trial-and-error learning mechanism to provide solutions, often outpacing traditional methods in terms of adaptability and optimization quality Zhou et al. (2019).

However, the application of DRL in real-world, mission-critical scenarios has been met with skepticism, primarily due to its inherent black-box nature. In many real-life operations, especially in contexts like port operations, the opaqueness of DRL's decision-making process can be a significant impediment. When DRL encounters unknown scenarios, its decision-making process can yield unpredictable outcomes. This unpredictability, combined with the lack of clear rationale behind decisions, is particularly concerning in environments where safety, reliability, and accountability are paramount. Decisions in high-stakes situations should ideally be transparent, traceable, adaptable, and readily understandable to human operators.

Recognizing this critical gap, we propose a novel approach, termed Deep Reinforcement Learning with Genetic Programming Hyper-Heuristic (DRL-GPHH). The fundamental premise behind DRL-GPHH is to synergistically combine the strengths of RL and Genetic Programming (GP) for truck dispatching. Specifically, the reinforcement learning network is trained to select from a repertoire of GP-generated heuristics. Each of these heuristics, rooted in the GP paradigm, inherently possesses a white-box characteristic, which ensures transparency and interpretability. Thus, even when the DRL chooses a particular heuristic, the ensuing decision is derived through the transparent and interpretable heuristic.

In this study, we employ RL as the high-level heuristic for selecting low-level heuristics, primarily for several compelling reasons. First, given that the truck dispatching problem is an NP-hard issue and is not amenable to solutions via supervised learning, RL stands out for its ability to learn the correct solution iteratively without needing pre-defined solutions. This characteristic is particularly advantageous in navigating the complex decision-making required in truck dispatching. Secondly, the DRL-GPEHH method introduced in this chapter necessitates dynamically generating continuous weights for various low-level heuristics. RL, capable of producing continuous values based on distribution, is inherently more suited to this requirement. Lastly, as highlighted in the work of Zhang et al. (2021b), RL is chosen for its demonstrated performance and training stability. It has been successfully applied across different domains, including edge computing (Chen et al., 2018c) and recommendation systems (Zheng et al., 2018), showcasing its versatility and effectiveness in addressing various problems.

The amalgamation of DRL's adaptability and GP's transparency paves the way for a method that strikes a harmonious balance between optimization efficiency and decision interpretability. Such a method not only retains the optimization prowess of DRL but also ensures that every decision is underpinned by a discernible rationale, courtesy of the GP heuristics. In essence, DRL-GPHH has the potential to be a game-changer for real-world port operation scenarios, ensuring both operational efficiency and decision-making transparency.

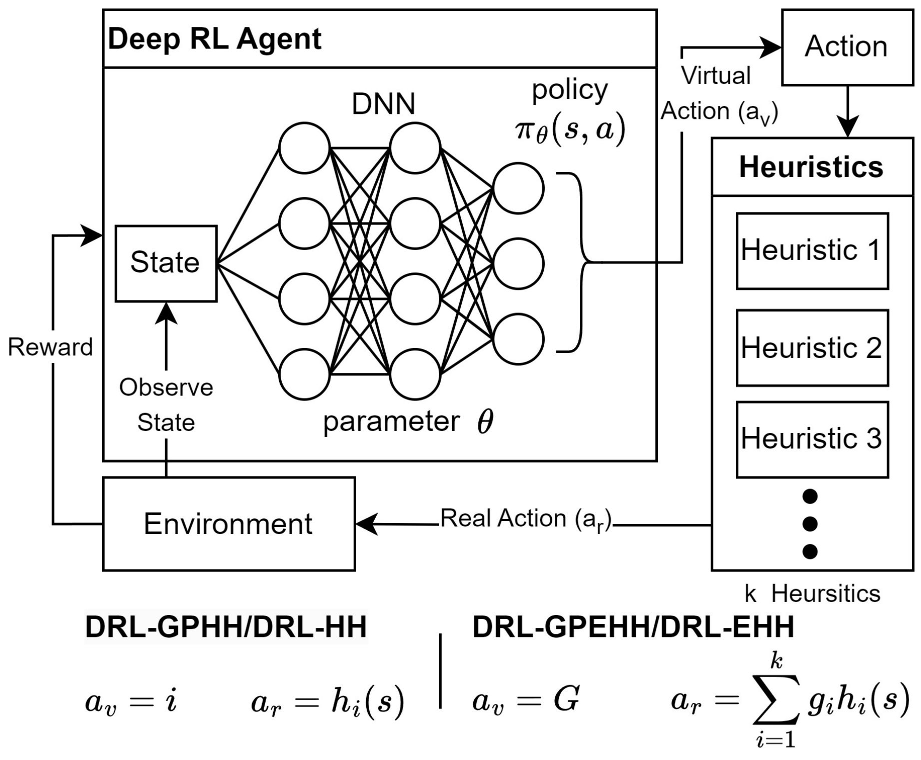


Figure 4.1: DRL-HH/DRL-GPHH & DRL-GPEHH/DRL-EHH Framework

As illustrated in Fig. 4.1, in the context of reinforcement learning genetic programming hyper-heuristics, given the `state` of an environment, the algorithm selects an appropriate `virtual action` according to a specific `policy`. This action subsequently picks a heuristic. Upon executing the heuristic, the actual action alters the environment, leading to a transition to a new state, denoted as S'. With each action execution, the algorithm receives a `reward` value and the estimated quality of the new state. The algorithm then adjusts its policy based on the magnitude of the reward value, ultimately maximizing the sum of rewards obtained when all steps are completed and the state reaches the `terminal` state.

In contrast to DRL-HH, which relies heavily on experts' domain knowledge for task scheduling and compromises its performance and generalization, DRL-GPHH replaces expert manual heuristics with GP-generated heuristics. In experiments, DRL-GPHH outperformed DRL-HH, highlighting the potential for collaborative problem-solving between RL and GP. Furthermore, given that GP-generated heuristics do not require the input of experts with extensive knowledge of the problem, they present a scalable solution for a range of similar complex optimization problems.

Specifically, DRL-GPHH and DRL-HH in this paper follow all the DRL settings in previous work Zhang et al. (2022), which adopts the double deep Q-learning (DDQN) algorithm Van Hasselt et al. (2016) as a high-level heuristic and uses a four-layer DRL network. The number of neurons in each layer is set as follows: 80, 100, 180, and 10, respectively. Rectified linear unit (ReLU) functions serve as the activation functions for each hidden layer, and the learning rate is set at 0.001.

The details of DRL-GPHH and DRL-HH are as follows:

4.1.1 Environment

The above-mentioned event-based port simulator serves as the training environment for DRL-GPHH and DRL-HH. This simulator uses the map and historical data of the Ningbo Meishan port that we cooperate with to simulate the real-world port. The simulator inputs the current state (S) into DRL-HH, and after the action (assignment of the truck) made by DRL-HH, it will deduce according to the rules and historical data to obtain the subsequent state (S'). While the simulator is running, various metrics are calculated, which are used as rewards to assist the training of DRL-HH.

4.1.2 State

The state, a set of matrices, represents the current operating environment in which the DRL-GPHH and DRL-HH will learn to select distinct actions depending on the state. In this study, the state is generated based on the trucks requiring task assignment, the current tasks and queuing statuses of QC) and YCs, the QC type and working status, the remaining number of tasks, and the average operating time of the cranes. The state matrix is of dimensions $i * j$, where i denotes the number of parameters incorporated within the state to describe a QC status, and the number of QC is represented by j. The specific parameters include:

- The QC remaining task number.
- The QC available task number.
- The QC bounded trucks number.
- The QC working status: 0 for unload and 1 for load.
- The QC type: 0 for standard and 1 for remote control.
- The minimum truck move time to a task start crane.
- The minimum waiting truck number of beginning cranes.
- The minimum waiting truck number of ending cranes.
- The average loading time of the beginning cranes.
- The average unloading time of the ending cranes.

It is important to note that since this truck dispatching problem in the container port is an optimization problem involving uncertainty, the state transition in this problem does not follow the conventional $s' = E(s, a)$ form. Instead, an uncertain parameter u is introduced, resulting in $s' = E(s, a, u)$. In this case, a represents the action chosen by the low-level heuristic selected by DRL-HH, while u is obtained during the simulator run.

4.1.3 Actions

DRL-GPHH and DRL-HH agents produce a virtual action to select a low-level heuristic, generating a real action for interacting with the environment.

DRL-GPHH uses the GP-generated low-level heuristics. Genetic programming was employed to learn and generate 10 distinct heuristics on training datasets across different scenarios. Each heuristic can manage the scenarios it has been trained on.

DRL-HH uses the manually designed low-level heuristics. These manual heuristics consider the distance from the task's starting point to the crane, the uniformity of the workload distribution among QC job lists, and the task's urgency. For each indicator, three thresholds are designed, resulting in nine manual heuristics. In conjunction with the aforementioned expert-designed manual heuristic 1.2.4, there are 10 heuristics available for selection.

4.1.4 Reward

Concerning rewards, the design for the DRL-GPHH and DRL-HH rewards adheres to the approach employed in the previous study Zhang et al. (2022), using the idle time of QCs as the primary reward component. For each assigned task i, its reward r_i is computed as $r_i = e_{i-1} - s_i$. The objective is to enhance port operational efficiency and reduce QC idle time; thus, the reward is a negative number, indicating that the smaller the QC idle time, the greater the reward. Moreover, since the QC idle time cannot be calculated when the action is executed, it must be updated after completion, requiring intermittent reward calculations. This process involves episodically computing the rewards for previous actions upon completing each task.

4.2 DRL-GPEHH & DRL-EHH

Although DRL-GPHH has demonstrated promising performance, it exhibits poor generalization when dealing with real historical data. Specifically, its performance deteriorates when encountering previously unseen data during training. This occurs because each scheduling operation in DRL-GPHH utilizes only one GP heuristic, with each heuristic incorporating information from the trained scenarios. Optimal performance is achieved only when the current scenario closely resembles one of the trained scenarios. However, the unselected heuristics in DRL-GPHH also encompass valuable information. By judiciously combining this information, the algorithm can adapt to a broader array of unseen scenarios and enhance performance.

As illustrated in Fig. 4.2, single-choice hyper-heuristics approaches (e.g., DRL-HH, DRL-GPHH) enable the high-level heuristic to select only one low-level heuristic for a given situation. For instance, if the high-level heuristic prioritizes task urgency, other factors, such as task proximity and node busyness, are consequently disregarded. In contrast, the ensemble hyper-heuristics frameworks proposed in this paper, namely DRL-EHH and DRL-GPEHH, dynamically allocate different weights to various low-level heuristics depending on the situation. This allows for a more nuanced consideration of multiple factors based on their assigned weights, including task proximity, node busyness, and task urgency. As a result, these ensemble frameworks make more accurate judgments and perform better in handling multi-scenario, complex, real-world optimization problems, as evidenced by our experimental results.

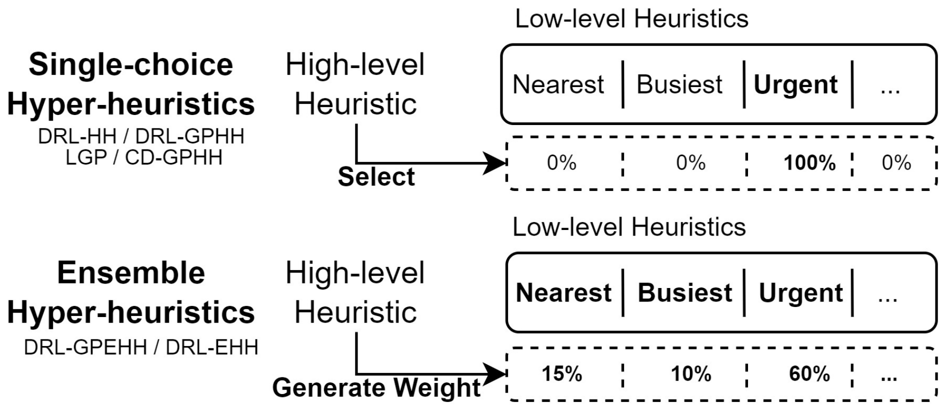


Figure 4.2: Single-choice Hyper-heuristic vs Ensemble Hyper-heuristic

Our experiments tested various methods for integrating multiple GP heuristics, as presented in Section 4.3. We found that the best-performing approach was to use DRL to adjust the weights of each GP heuristic continuously. As illustrated in Fig. 4.1, DRL-GPEHH employs a reinforcement learning agent as a gating mechanism to assign weights to multiple heuristic experts, subsequently combining the results of these heuristics to produce the final assignment. The DRL-GPEHH offers several advantages over DRL-GPHH primarily due to the integration of multiple expert heuristics, which leads to improved decision-making and adaptability. The key benefits include:

- Diversity and adaptability: DRL-GPEHH incorporates various GP heuristics, each with strengths and weaknesses. This diversity allows the algorithm to adapt to different situations and select the most suitable heuristic for a given scenario, leading to better overall performance and leveraging the strengths of other heuristics to compensate for their limitations.

- Learning efficiency: By utilizing the knowledge and experience embedded in multiple heuristics, DRL-GPEHH can potentially accelerate the learning process. As a result, the algorithm can converge to a near-optimal solution more quickly than a normal DRL-HH, which relies solely on its learning and exploration.
- Knowledge transfer: DRL-GPEHH can benefit from knowledge transfer between the heuristics, allowing the algorithm to capitalize on their combined knowledge. This leads to more effective exploration and exploitation strategies, ultimately improving the quality of the solutions.
- Scalability: The ensemble approach enables DRL-HH to handle a broader range of problems and larger-scale instances. By combining the expertise of multiple heuristics, the algorithm can scale better to tackle complex tasks and adapt to new, unseen scenarios.
- Algorithm automation: The utilization of GP-generated heuristics obviates the need for expert inputs, thereby significantly enhancing the automaticity of algorithm generation. This culminates in an efficient and self-reliant design process. This heightened level of automation simplifies the task of addressing complex real-world problems, thereby extending the versatility and applicability of the method.

In summary, DRL-GPEHH outperforms DRL-GPHH by leveraging multiple GP heuristics' strengths, improving adaptability, robustness, learning efficiency, knowledge transfer, and scalability. These advantages make it more suitable for solving complex and dynamic problems like container terminal truck dispatching in various domains.

Furthermore, to facilitate a more precise comparison of the performance of DRL-HH, DRL-GPHH, and DRL-GPEHH, DRL-EHH is proposed to act as a control. DRL-EHH employs the same manual heuristics as those in DRL-HH, replacing the GP-generated low-level heuristics in DRL-GPEHH. Except for the low-level heuristics, all settings of DRL-EHH are identical to those of DRL-GPEHH. Consequently, we only describe the structure of DRL-GPEHH.

To handle multi-dimensional actions and output continuous weights for multiple heuristics simultaneously, DRL-GPEHH employs a multi-action proximal policy optimization (PPO) as a gate agent instead of the double deep Q-Network (DDQN) used in DRL-GPHH.

PPO is an advanced policy optimization algorithm introduced in 2017 Schulman et al. (2017) designed to overcome the challenges faced by other algorithms, such as trust region policy optimization (TRPO) and asynchronous advantage actor-critic (A3C). It enhances sample efficiency and stability by utilizing a trust region approach and employing a clipped objective function shown in (4.1). Here, $r_t(\theta)$ is the probability ratio between the current policy and the old policy, represented as $\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$. The variable $\hat{A}_t$ denotes the estimated advantage function at time step t, and ϵ is a hyperparameter controlling the degree of trust region in the policy update.

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t\left[\min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)\right] \tag{4.1}$$

Combined with the formula, the training process of the multi-action PPO is described in Algorithm 3.

To maintain fair competition, DRL-GPEHH retains the deep neural network (DNN) settings of DRL-GPHH. However, it doubles the networks to

Algorithm 3 Proximal Policy Optimization (PPO) Training

```
 1: Initialize policy parameters $\theta$ and value function parameters $\phi$
 2: for each iteration do
 3:     Collect a set of trajectories $\tau$ using the current policy $\pi_\theta$
 4:     Compute rewards-to-go $R_t$ for each time step in trajectories
 5:     Compute advantage estimates $A_t$ using value function $V_\phi$
 6:     for each optimization epoch do
 7:         for each time step $t$ in trajectories do
 8:             Compute probability ratio $\rho_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$
 9:             Compute surrogate objective $L_t(\theta) = \min(\rho_t(\theta)A_t, \text{clip}(\rho_t(\theta), 1-\epsilon, 1+\epsilon)A_t)$
10:             Perform gradient ascent on $\theta$ to maximize $E_t[L_t(\theta)]$
11:             Update value function parameters $\phi$ by minimizing the value loss
12:         end for
13:     end for
14:     Update policy $\pi_{\theta_{\text{old}}} \leftarrow \pi_\theta$
15: end for
```

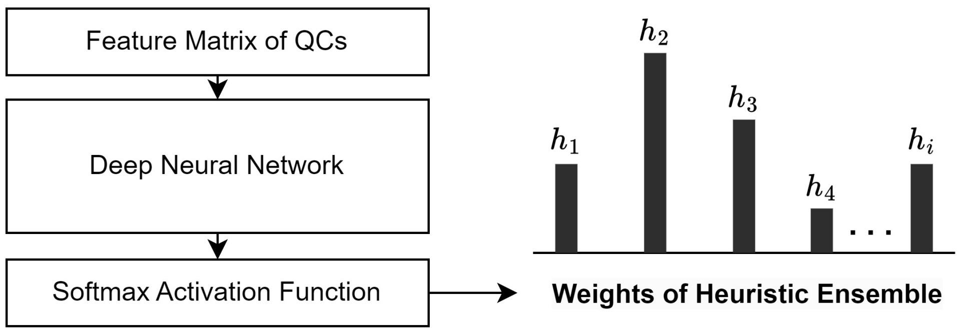


Figure 4.3: Neural Network Output of DRL-GPEHH and DRL-EHH

serve as a policy network and a value network, respectively. Furthermore, a softmax activation function is incorporated following the DNN output. As illustrated in Fig. 4.3, DRL-GPEHH produces a continuous weight vector for each heuristic. Based on these weights and the outputs of the heuristics, DRL-GPEHH synthesizes the final real action for dispatching the truck.

The environment and status settings of DRL-GPEHH are consistent with DRL-HH, but some modifications have been made in the following parts:

4.2.1 Actions

Similar to DRL-GPHH, DRL-GPEHH does not directly interact with the environment through actions as depicted in Fig. 4.1. Instead, it employs the same GP-generated heuristics used in DRL-GPHH as low-level heuristics. The distinction between the two lies in their utilization of low-level heuristics: in DRL-GPEHH, instead of directly selecting the most appropriate task for output as in DRL-GPHH, the low-level heuristics generate a task ranking based on their internal rules.

Algorithm 4 Weighted Ensemble Ranking

```
Require: $tasks, heuristics, weights$
Ensure: $best\_task$
1: $scores \leftarrow \{ta : 0 \mid ta \in tasks\}$
2: for $h, w$ in $heuristics, weights$ do
3:     $ranked\_tasks \leftarrow h.rank(tasks)$
4:     for $i, t$ in enumerate($ranked\_tasks$) do
5:         $scores[t] \leftarrow scores[t] + (i + 1) * w$
6:     end for
7: end for
8: $best\_task \leftarrow \arg\min_{t \in tasks} scores[t]$ return $best\_task$
```

Based on the rankings of available tasks generated by heuristics, DRL-GPEHH employs the weighted ensemble ranking method, as illustrated in Algorithm 4, to combine the ranks and determine the best task. This algorithm takes a set of tasks, an array of heuristics, and their respective weights as input, with the objective of identifying the optimal task from the available options.

4.2.2 Reward

Given that the action space of DRL-GPEHH is much larger than that of DRL-GPHH, and considering the challenges introduced by delayed rewards in terms of temporal credit assignment, exploration, and convergence, the rewards employed in DRL-GPHH become less suitable for DRL-GPEHH. Specifically, the temporal credit assignment problem arises due to the difficulty associating the correct action with an observed reward when rewards are delayed. This can slow the learning process or cause the agent to learn suboptimal policies. Additionally, delayed rewards can impact the exploration-exploitation trade-off, as the agent may need to explore the environment extensively before discovering the long-term consequences of its actions, potentially delaying convergence to an optimal policy.

To address these challenges, we introduced a new reward function that combines reward shaping and imitation learning to enhance the learning process in the presence of delayed rewards by improving credit assignment, encouraging efficient exploration, and stabilizing convergence. The new rewards are designed as $reward = e_{i-1} - s_i - \delta cov(O_r, O_m)$, where δ is the weight, cov is the covariance calculation function, O_r is the task ranking given by DRL-GPEHH, and O_m is the ranking given by manual heuristics described in Section 1.2.4.

The reward for rankings similar to the manual heuristic can be adjusted by setting different weight values, denoted by δ. In this paper, δ is set to

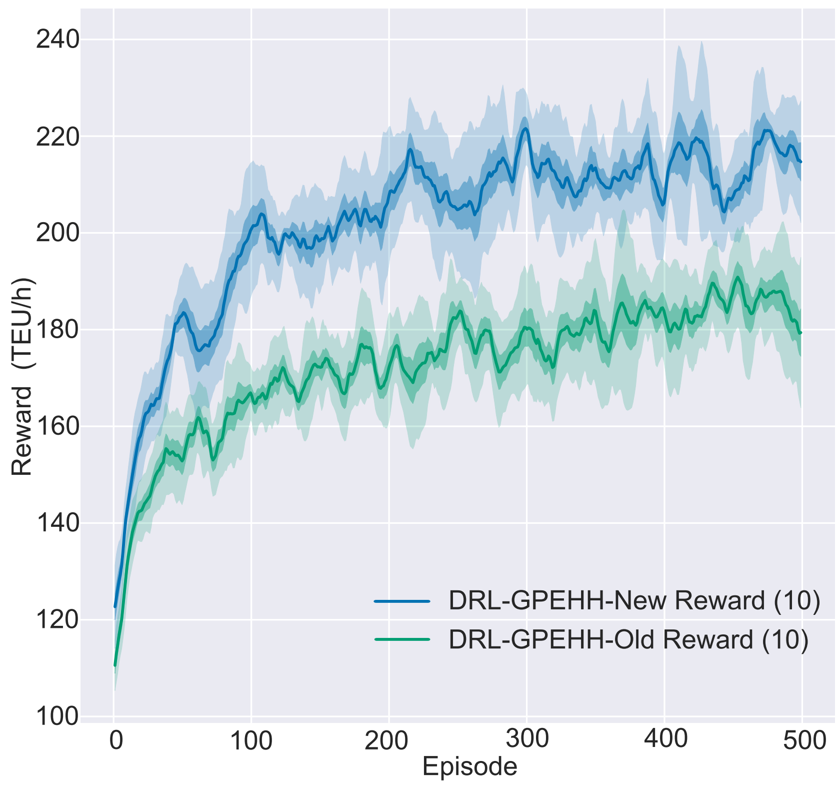


Figure 4.4: Performance of DRL-GPEHH with Varying Reward Structures

κ/en, where κ is a scaling factor that can be adjusted according to the size of the previous reward term, which is set to 10 in this study, and en represents the number of training episodes. The weight of this reward gradually decreases as the number of training generations increases, encouraging the algorithm to learn from the manual expert heuristic initially and reducing the influence of convergence to the manual heuristic on the algorithm's ability to reach a superior solution during later stages of learning.

Although the reward component of $e_{i-1} - s_i$ in DRL-GPEHH is the same as that in DRL-HH, it must be calculated after the task completion, while the $-\delta cov(O_r, O_m)$ component can be obtained immediately, addressing the reward delay issue. In our newly designed reward structure, we guide the DRL-GPEHH to learn like the manual heuristic by encouraging reward allocations that resemble task rankings generated by the manual heuristic. According to the experimental results in the training datasets detailed in Section 4.3 with 10 random seeds shown in Fig. 4.4, the newly designed rewards indeed achieve better performance, speed up convergence, and resolve the problem of delayed rewards in container terminal truck dispatching problem.

4.3 Experiments and Discussion

In the ensuing section, we undertake a comprehensive evaluation of DRL-GPHH and DRL-GPEHH, focusing on a multifaceted container port dispatching problem marked by uncertain parameters. This analysis is positioned against DRL-HH and DRL-EHH to elucidate the advantages of integrating GP with RL. Given the proven robustness and reliability of the manually delineated heuristic outlined in Section 1.2.4, which has demonstrated considerable efficacy in practical port applications, we employ it as a benchmark baseline. Thus, all ensuing comparative analyses are premised on enhancements made relative to this manual heuristic (Imp.). Additionally, this section incorporates ablation studies and sensitivity analysis, furnishing insights into the conducive elements underpinning the exceptional performance of DRL-GPEHH.

4.3.1 Experiment Design

As this work aims to develop an algorithm that can be deployed in a real-world port to enhance its operational efficiency, all data used in the experiments are derived from actual historical operating data of Ningbo Meishan Port, with which we collaborate. We sampled 20 days of operation data to generate 10 training sets and 10 test sets, each containing 4,000 tasks. All experiments are conducted using the event-driven simulator we developed, with simulator parameters adjusted based on historical operating data. There are two ship berths 10 operating QCs, and the number of container trucks varies between 50 and 80. Moreover, as mentioned in Section 1.2.3, the operating times for QC and YC are derived from real-world historical data, while the truck travel times are drawn from the historical time distribution.

We trained 10 GPs for 300 generations with 100 different random seeds on the same set of 10 training datasets and selected one best-performing individual from each dataset to form 10 GP-generated low-level heuristics. The GP algorithm and parameters are consistent with the GP algorithm with logic operators (LGP) described in our previous paper Chen et al. (2022), featuring a population size of 1024 and crossover, mutation, and reproduction rates of 60%, 30%, and 10%, respectively. All algorithms were executed 100 times with different random seeds, and the training phase consisted of 1000 episodes. Subsequently, we assessed the performance of DRL-HH Zhang et al. (2022), DRL-GPHH, DRL-EHH, and DRL-GPEHH on the test sets.The average results of the training are presented in Table 4.1.

4.3.2 Experiment Results

The experimental results demonstrate that, regardless of whether using DRL to select single heuristics or a set of heuristics, GP-generated heuristics outperform manually-designed heuristics. Replacing human-designed low-level heuristics in DRL-HH with GP-generated heuristics provides a performance boost of 1.42%, while DRL-GPEHH achieves a 9.88% improvement over DRL-EHH when multiple GP heuristics are jointly involved in decision making as an ensemble at each step. It is worth noting that due to the theoretical optimization upper bound in the port truck dispatch problem, the closer the TEU/h is to the upper limit, the more difficult it becomes to improve performance further. The 17.77% performance improvement of DRL-GPEHH over manual heuristics demonstrates its effective combination of knowledge from multiple GP-generated heuristics. Unlike DRL-GPHH, although both use the same 10 heuristics, the ensemble-based approach leverages the knowledge from multiple GP-generated heuristics to achieve better performance.

Table 4.1: DRL-HH, DRL-GPHH, DRL-EHH and DRL-GPEHH Training Results (TEU/h)

No.	**Manual**	**DRL HH**	**DRL GPHH**[1]	**DRL EHH**	**DRL GPEHH**[1]
1	202.36	222.91	225.50	218.18	240.75
2	188.54	205.56	204.56	201.46	225.10
3	182.31	199.26	201.73	198.14	212.46
4	191.33	205.66	207.86	204.54	219.60
5	186.26	196.52	204.75	202.54	221.28
6	190.69	204.51	211.14	205.85	220.87
7	193.33	202.14	204.17	208.60	234.28
8	191.59	203.24	198.27	203.33	224.02
9	186.14	198.19	204.97	203.05	216.79
10	190.98	204.97	206.91	208.05	226.54
Avg.	190.35	204.30	206.98	205.37	224.17
Imp.	**N.A.**	**7.32%**	**8.74%**	**7.89%**	**17.77%**

[1] DRL-GPHH and DRL-GPEHH significantly differ from other algorithms, $p < 0.05$.

In contrast to the excellent performance of the GP-generated heuristic ensemble, the application of a manual heuristic ensemble in DRL-EHH results in only a 0.57% performance improvement compared to DRL-HH. This can be attributed to the simplicity and high similarity across the adopted manual heuristics, which, while capable of producing satisfactory results for straightforward and popular scenarios, makes it challenging to improve performance further. Conversely, the GP-generated heuristic ensemble, owing to its complex internal structure and knowledge that encompasses various scenarios, has the potential to achieve superior performance. However, continuous weight adjustments are required in different environments to maximize the utilization of knowledge from multiple GP heuristics. The experiments supporting this statement and the reasons for the excellent performance of DRL-GPEHH will be presented in the subsequent subsection.

Next, we put the trained DRL-HH, DRL-GPHH, DRL-EHH, and DRL-GPEHH into a test environment completely different from the training environment with a broadly similar baseline.

As delineated in Table 4.2, the algorithms DRL-GPHH and DRL-GPEHH, which employ GP-generated low-level heuristics, outperform their manually designed heuristic counterparts, DRL-HH and DRL-EHH, by margins of 0.43% and 8.14%, respectively. This substantiates the notion that integrating DRL with GP-generated heuristic ensembles can yield substantial performance improvements, even in unfamiliar testing conditions. Moreover, DRL-EHH and DRL-GPEHH exhibit enhanced consistency in the test set by leveraging an ensemble of GP heuristics at each decision point. The performance decrement observed for these ensemble-based models on the test set is approximately 1% less than that for the non-ensemble alternatives. This finding not only reinforces the effectiveness of ensemble approaches in navigating unknown scenarios but also highlights the robustness and practical applicability of our proposed techniques, which demonstrate minimal performance attrition in dynamically changing real-world contexts.

Across both training and test sets, DRL-GPEHH outperforms all other al-

Table 4.2: DRL-HH, DRL-GPHH, DRL-EHH and DRL-GPEHH Test Results (TEU/h)

No.	**Manual**	**DRL HH**	**DRL GPHH**[1]	**DRL EHH**	**DRL GPEHH**[1]
1	190.48	199.74	203.41	215.88	227.40
2	203.54	209.55	210.20	203.01	231.24
3	189.82	195.91	205.08	203.83	217.97
4	187.04	198.26	195.73	202.24	213.35
5	191.43	202.51	202.96	199.57	216.45
6	182.90	193.37	192.97	203.92	207.67
7	193.62	204.61	201.89	202.78	221.44
8	185.86	194.05	193.95	201.60	212.82
9	191.29	200.59	203.29	198.85	217.13
10	188.87	195.22	192.55	199.95	221.30
Avg.	190.48	199.38	200.20	203.16	218.68
Imp.	**N.A.**	**4.67%**	**5.10%**	**6.66%**	**14.80%**
Dec.	N.A.	-2.65%	-3.64%	-1.24%	-2.96%

[1] DRL-GPHH and DRL-GPEHH significantly differ from other algorithms, $p < 0.05$.

gorithms, substantiating the synergistic potential between DRL and GP ensembles in augmenting algorithmic performance. This ensemble approach not only enhances generalization and robustness but also resolves the limitations of DRL-HH, which is overly reliant on the quality and diversity of pre-defined heuristics. Incorporating GPHH adds a new dimension of diversity, adaptability, and efficiency. Furthermore, by automating the generation of low-level heuristics through GP, both DRL-GPHH and DRL-GPEHH eliminate the need for manual expert design, thereby considerably increasing the level of automation in algorithm design for complex problems like marine port truck dispatching. This automated approach proves advantageous in adapting to various complex, real-world optimization challenges, offering a scalable, flexible, and robust solution across diverse problems.

Finally, it has been substantiated that DRL-GPEHH can yield superior results compared to DRL-HH, DRL-GPHH, DRL-EHH, and manual heuristics in container port truck dispatching. Although DRL-GPEHH has not yet been implemented in real-life port settings, our manually crafted heuristic algorithm, used as a baseline in this study, has been successfully utilized at Ningbo Port for several years. Statistical analysis conducted by the port reveals that work efficiency has increased by 8.1%, while ship docking time has decreased by 2.2%. This high-performing algorithm has resulted in time savings, facilitated the handling of more ships, and consequently, significantly enhanced the profitability of the port company. As a future endeavor, we plan to collaborate with relevant stakeholders to comprehensively evaluate and deploy the proposed DRL-GPEHH algorithm in real-world scenarios.

4.3.3 Ablation and Sensitivity Analysis

Subsequently, we extend our experimental investigation to corroborate the superior performance of the DRL-GPEHH. We formulate three hypotheses to guide this analysis: first, that standard RL algorithms struggle to converge in complex dynamic environments with extensive action spaces, thereby necessitating a hyper-heuristics framework; second, that basic GP hyper-heuristics are limited in their generalizability and capacity to handle intricate scenarios, thus impeding overall algorithmic performance on test sets; third, The optimal performance of the GP ensemble methodology is realized by dynamically assigning weights to GP individuals.

To empirically evaluate the performance of standard RL algorithms in complex test environments, we trained DDQN and PPO algorithms using training sets. The configurations for DDQN and PPO were strictly in line with established settings in the literature Van Hasselt et al. (2016); Schulman et al. (2017), and each was subjected to 1,000 training episodes. For a more comprehensive assessment, we also incorporated a random dispatching algorithm into our evaluation framework. As the test results presented in Table 4.3, it is unequivocally evident that DDQN failed to acquire any meaningful information during training, performing on par with randomized strategies. Similarly, PPO exhibited only marginal improvements over random dispatching and did not approach the effectiveness of heuristic methods. These experimental outcomes compellingly substantiate our first hypothesis: conventional RL algorithms face significant challenges in converging within complex and dynamic environments characterized by sparse rewards and expansive action spaces. This underscores the necessity for adopting a hyper-heuristics framework, where low-level heuristics replace traditional actions to stabilize the learning environment. Such an arrange-

Table 4.3: Performance of Other Algorithms (TEU/h)

	Perf.	**Manual**	**Random**	**DDQN**	**PPO**	**LGP**	**CD-GPHH**	**VMHE**	**RMHE**	**WRMHE**	**BGPE**	**VGPE**	**RGPE**	**WRGPE**
Train	**Avg.**	190.35	160.84	168.51	172.24	198.43	208.49	169.23	177.03	201.65	202.72	203.02	208.25	204.39
	Imp.	**N.A.**	**-15.50%**	**-11.47%**	**-9.51%**	**4.24%**	**9.53%**	**-11.09%**	**-7.00%**	**5.94%**	**6.50%**	**6.66%**	**9.40%**	**7.37%**
Test	**Avg.**	190.48	161.28	161.35	167.84	194.26	198.21	169.64	176.71	196.66	197.13	198.92	204.25	203.54
	Imp.	**N.A.**	**-15.33%**	**-15.29%**	**-11.89%**	**1.98%**	**4.06%**	**-10.94%**	**-7.23%**	**3.24%**	**3.49%**	**4.43%**	**7.22%**	**6.86%**

Table 4.4: Standard Deviation of Action Select Ratio in DRL-EHH and DRL-GPEHH

	Act 1	**Act 2**	**Act 3**	**Act 4**	**Act 5**	**Act 6**	**Act 7**	**Act 8**	**Act 9**	**Act 10**	**Avg.**
DRL-EHH	5.03%	5.10%	0.62%	3.13%	0.23%	0.99%	3.43%	0.03%	0.00%	0.22%	**1.88%**
DRL-GPEHH	3.81%	6.84%	6.75%	7.40%	7.64%	6.90%	6.25%	4.22%	5.92%	8.94%	**6.47%**

ment permits RL algorithms to assimilate valuable information and exhibit improved performance, as the DRL-HH model exemplifies.

Based on the second hypothesis, we trained two distinct populations using LGP and Cooperative Double-Layer Genetic Programming hyper-heuristics (CD-GPHH) Chen et al. (2022). Both approaches employ a hyper-heuristics framework that leverages GP as a high-level selector for low-level GP heuristics. The configurations for LGP and CD-GPHH were consistent with those outlined in our prior work Chen et al. (2022). During training, each population was exposed to a sequence of 1000 generations across multiple training sets, each initialized with 100 different random seeds. The best-performing individuals were then selected for further analysis. Evaluation of the training set yielded average performance improvements over the manual heuristic of 4.24% and 9.53% for LGP and CD-GPHH, respectively. However, the corresponding gains on the test set were comparatively modest: 1.98% and 4.06%, as shown in Figure 4.3. Although CD-GPHH outperformed LGP due to its double-layer architecture, it fell short of the 15% improvement observed in smaller test datasets from our previous study Chen et al. (2022). The diminished performance on the test set underscores the limitations of relying solely on GP hyper-heuristics for complex optimization tasks in large-scale, multi-scenario environments. This substantiates our decision to integrate RL with heuristic ensembles for tackling real-world, large-scale challenges.

To further validate our second hypothesis, we implemented three distinct ensemble methods—voting, ranking, and weighted ranking—to amalgamate manual and GP heuristics. The effectiveness of these methods was then assessed using the test set delineated in Section 4.3. Each heuristic selects an optimal task in the voting mechanism, with the majority vote determining the final output. Ranking involves each heuristic assigning a rank to all tasks, culminating in the task with the highest aggregated rank chosen as the output. The weighted ranking method extends this by applying weights to the summed ranks according to Algorithm 4. For comparative analysis, we also introduced the Best GP Ensemble (BGPE) method, which involves assessing 10 unique GP low-level heuristics on each dataset and choosing the most productive one.

Then, to get the weight value used in the weighted ranking method, we calculated the probabilities of DRL-HH and DRL-GPHH, selecting each heuristic, as depicted in Figure 4.5. For manual heuristics, DRL methods preferred one or two specific actions, such as Act1 and Act2, while less frequently opting for actions like Act8, Act9, and Act10. In contrast, the selection probabilities for GP heuristics were more evenly distributed, with no single heuristic predominating. This suggests that while DRL methods gravitate towards better-performing manual heuristics, the uniform performance exhibited by GP heuristics makes it challenging for DRL-GPHH to identify a singular, superior heuristic. Such findings imply that the GP ensemble harbors a wealth of knowledge, making optimally assigning fixed weights nontrivial.

The inferior performance of the vote manual heuristic ensemble (VMHE) and rank manual heuristic ensemble (RMHE) in Table 4.3, compared to the expert manual heuristic, indicates that when multiple manual heuristics make decisions, the subpar performance of certain heuristics adversely affects the overall decision-making performance. This leads to a final performance that is 10.94% and 7.23% worse than the individual expert manual heuristics in the manual heuristic ensemble, respectively. Moreover, when utilizing the statistical DRL probabilities to select each action shown in Fig. 4.5 as weights for the weighted rank manual heuristic ensemble (WRMHE) method, it achieves performance comparable to DRL-GPHH. This further

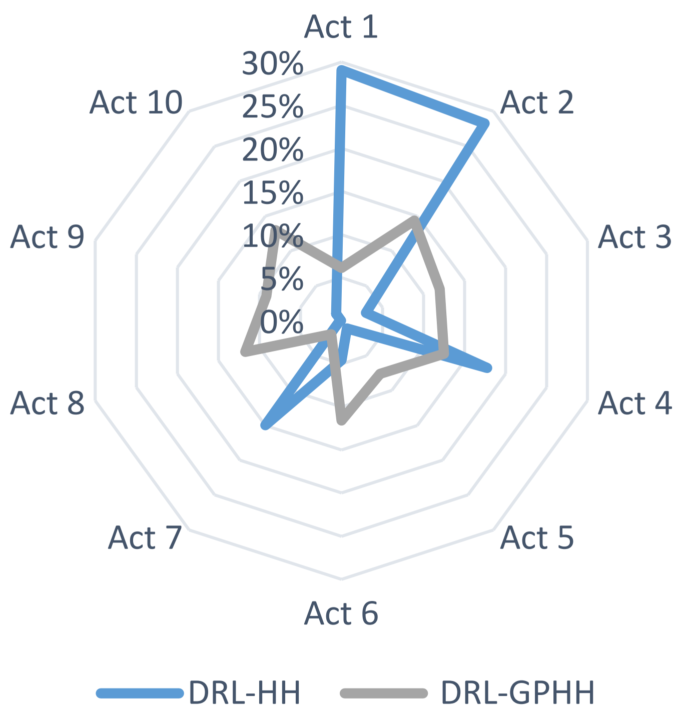


Figure 4.5: DRL-HH/DRL-GPHH Action Selection Ratio

underscores the significance of adjusting the weights of heuristic ensembles.

However, as demonstrated in Table 4.3, we find that the vote GP ensemble (VGPE) and rank GP ensemble (RGPE) methods, which use the GP heuristic ensemble, perform significantly better than the best GP ensemble (BGPE) method that relies on a single heuristic. Nevertheless, the weighted rank GP ensemble (WRGPE) method, which utilizes the proportions of different actions selected by DRL-GPHH as weights, does not improve performance compared to RGPE. The performance of WRGPE is worse than that of the unweighted RGPE. This observation supports the argument that applying fixed weights to the GP heuristic ensemble does not enhance performance and may prove detrimental.

Furthermore, we calculated the standard deviations of the weights assigned to different actions by DRL-EHH and DRL-GPEHH. As shown in Table 4.4, the standard deviation of DRL-EHH is 1.38%, indicating that for manual heuristics, DRL-EHH tends to assign relatively consistent weights to each action. In contrast, the standard deviation of DRL-GPEHH is 6.47%, illustrating that the weights of each action in DRL-GPEHH vary during each scheduling, contributing to its superior performance.

These supplementary experiments substantiate our initial hypotheses: Conventional RL methods struggle with large-scale real-world problems and thus necessitate integrating a hyper-heuristics framework. Likewise, simplistic GP hyper-heuristics approaches fail to formulate a universally applicable model capable of managing large-scale, real-world scenarios. However, DRL-GPEHH by continually fine-tuning the weights of diverse low-level GP heuristics during the decision-making process—effectively capitalizes on the rich knowledge reservoir inherent in the GP heuristic ensemble. This leads to marked enhancements in performance, stability, and generalization capabilities.

4.4 Summary

This chapter has delved into the intricate realm of dynamic truck dispatching in container terminals and has successfully introduced two pioneering methodologies—DRL-GPHH and DRL-GPEHH, both of which employ learning-assisted genetic programming. Among the duo, the DRL-GPEHH stands out, ingeniously amalgamating an ensemble of GP heuristics, thus bridging the gaps present in DRL-HH and DRL-GPHH methodologies.

One of the most salient achievements of DRL-GPEHH lies in its ability to generate heuristics autonomously. This eliminates the heavy dependence on expert intervention, fostering greater adaptability to a myriad of operational environments and propelling the field of algorithmic development toward a higher echelon of automation. Such a transition is crucial, especially in a dynamic domain like truck dispatching in container terminals, which demands flexibility and accuracy.

A meticulous experimental evaluation has further validated the supremacy of the DRL-GPEHH approach. With its stellar performance across pivotal metrics such as dispatching efficiency, adaptability, and generalization, DRL-GPEHH has set new benchmarks. The embodiment of a deep reinforcement learning agent as a sentinel, meticulously distributing weights to individual GP heuristics, symbolizes an advanced phase in algorithmic design. By facilitating a synergistic interaction between the ensemble of GP heuristics and DRL, the system guarantees optimal task assignments. This symbiotic relationship, enhanced by an innovative action-reward schema, ensures swift convergence, bestowing the algorithm with augmented stability and superior generalization.

This chapter's revelations underscore the immense potential of Genetic Programming, particularly with its ensemble counterparts, within the overarching framework of reinforcement learning hyper-heuristics. Such methodologies stand poised to revolutionize complex real-world optimization problems. As we gaze into the future, we envision extensive exploration into collaborative training mechanisms involving GP and DRL. Such endeavors promise not only to refine performance metrics but also to broaden the applicability of the DRL-GPEHH model, ushering it into a wider array of domains and thereby redefining the boundaries of optimization.

Chapter 5

Machine Learning Assisted Methods in Dispatching Strategy Evaluation Accuracy Enhancing

While we have previously discussed various methods to enhance truck dispatching efficiency, it is crucial to highlight that their testing and training were rooted in traditional event-based simulators. Nevertheless, an inherent limitation has surfaced in these simulators: they frequently neglect certain specific route or junction-related regulations, leading to overly optimistic performance estimates. An in-depth comparison between authentic port operation data and these simulated results unveiled a notable disparity. In particular, the time trucks spend at intersections, which tends to be overlooked in event-based simulations, was consistently underestimated. In such simulations, trucks are seamlessly transitioned from one crane to another without accounting for potential delays at intersections.

To address this issue, we developed a time-stepped simulation that tracks truck actions and interactions at every time step. This approach allows for a comprehensive tracking of trucks' positions, resulting in a more accurate performance estimation. However, time-stepped simulations pose a dilemma concerning the setting of the time step. If the time step is excessively large, the truck route simulation will lack precision. Conversely, if the time step is too small, the computational cost becomes prohibitively high. Our findings reveal that a time-stepped simulator with a one-second time step is approximately 200 times more computationally intensive than an event-based simulation in the context of container truck dispatching. The magnitude of this computational overhead primarily stems from the need to accurately assess a truck's passage through an intersection, which necessitates the evaluation of the truck's collision relationship at each time step. This computational cost is particularly prohibitive when training auto-generated truck dispatching strategies, which often require tens of thousands of simulation runs (Chen et al., 2020).

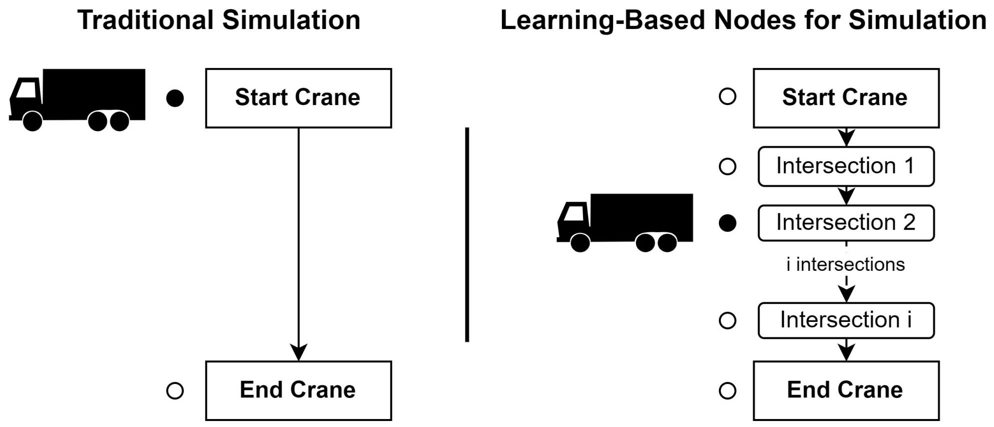


Figure 5.1: The Difference Between Traditional and Proposed Simulation

To reconcile the need for controlled simulation accuracy with computational efficiency, we introduced intersection nodes into the event-based simulation model. This modified framework enables trucks to move from node to node instead of directly from crane to crane (Fig. 5.1). Intelligent intersection nodes automatically estimate the passing time of trucks at intersections based on learned data from previous truck movements and current environment status.

Despite this advancement, a new challenge emerged. Most ports lack precise Global Positioning System (Global Positioning System (GPS)) data for trucks, and when available, the data is often unreliable and requires substantial pre-processing. To overcome this, we introduced learning-based methods to infer truck movements at intersections using existing port operation data.

This chapter presents a comparison of two such learning-based methods - Genetic Programming and Reinforcement Learning - to generate estimates of truck passing times at intersections. Our findings reveal that while RL outperforms GP in accuracy, it is also significantly more computationally intensive. To optimize this trade-off, we propose a hybrid method combining the advantages of GP and RL, augmented by an intersection importance analysis framework. Utilizing insights derived from data, we rank the influence of all intersections on simulation accuracy. Consequently, we designate the more critical intersections to be controlled by RL, while GP manages the remainder.

This innovative approach successfully strikes a balance between performance and computational cost. Furthermore, it enables us to generate more precise performance estimates and develop more efficient truck dispatching strategies. Importantly, our method can be readily applied to other transportation simulation challenges, significantly improving simulation accuracy even without detailed GPS location and other precise operation data.

This chapter delineates key advancements in enhancing the precision of event-based simulations for truck dispatching in container ports. The primary content can be summarized as:

- We introduce learning based surrogate nodes in event-based container port truck dispatching simulations. This novel approach improves simulation accuracy without significantly increasing computational cost, addressing a key limitation of traditional event-based simulations.
- We propose the integration of GP, RL, and a novel GP and RL hybrid (Genetic Programming and Reinforcement Learning Hybrid (GPRL-H)) method to simulate truck actions at these intersections. The hybrid method provides a preferable balance between simulation accuracy and computational cost.
- We develop a method for enhancing the accuracy of truck dispatching simulations in the absence of precise GPS location data. This approach opens avenues for increased simulation precision in settings without precise data.
- We introduce a data-driven framework for analyzing the importance of intersections. This innovative approach ranks intersections based on their significance, facilitating more effective control over intersection simulations.
- We provide empirical evidence demonstrating the superior performance of our proposed methods. Through rigorous experimentation, we show that our new hybrid method achieves an excellent performance.

The rest of this chapter is organized as follows. Section 5.1 and 5.2 introduces the dynamic truck dispatching and the simulation problem. The proposed learning-based methods are delineated in Section 5.4, 5.5 and 5.6. Section 5.7 presents the experimental results and offers insights into the strong performance of our new framework. Finally, conclusions are drawn in Section 5.8.

5.1 Truck Dispatching Simulation Problem

Given the critical role of simulation in container truck dispatching optimization within container ports, numerous researchers have employed simulation-supported optimization methods to address this problem (Tao and Qiu, 2015; Rajendran, 2021; Mirzaei-Nasirabad et al., 2023). Most, however, have favored the use of event-based simulations. While these simulations offer simplicity and speed, a significant drawback is their neglect of trucks traveling complexities on the road.

Traditional event-based simulations typically calculate the distance between the start and end positions and then divide this distance by a fixed or fluctuating speed to estimate the travel time (Juan et al., 2013). As such, they overlook the actions occurring during the travel process, particularly the interactions of trucks at intersections, which heavily influence the accuracy of the final simulation outcome (Arvin et al., 2020).

Fig. 5.2 illustrates two typical situations: when a truck aims to cross an intersection and another truck is already present from a perpendicular direction, the first truck must wait. And even when there's no truck from a perpendicular direction, if a truck is approaching from the opposite direction, the initial truck intending to turn left must wait. These situations only consider two trucks, but the complexity increases with more trucks involved.

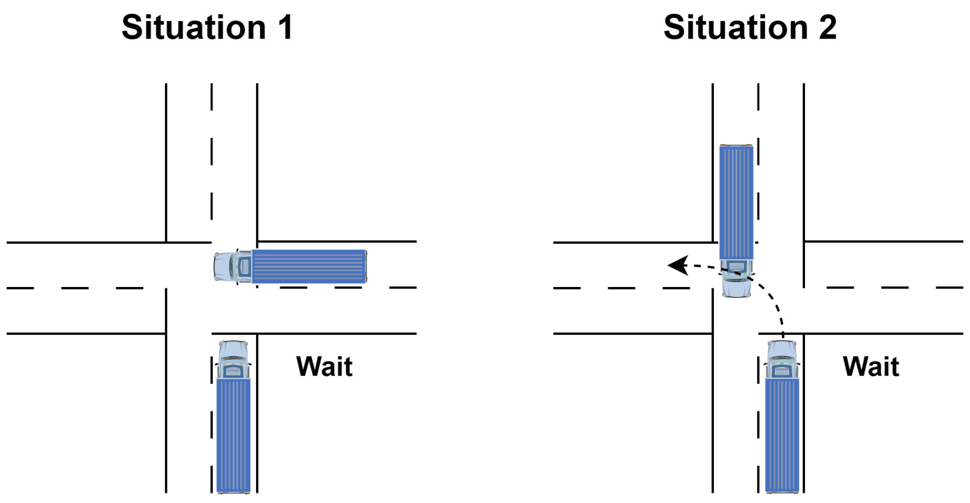


Figure 5.2: Examples of Intersection Actions

This complexity is crucial in container port truck dispatching simulations and should not be overlooked. To simulate truck routes and actions at intersections precisely, we would ideally employ time-stepped simulation instead of event-based simulation to track truck statuses at each time point (Gould et al., 2007). Regrettably, very few studies have used time-stepped simulation when optimizing truck dispatching due to its high computational cost and its difficulty in algorithm training and strategy optimization.

This challenge was the impetus for introducing learning-based methods in event-based simulation to simulate intersection actions. Our approach aims to enhance accuracy by accurately simulating intersection behaviors, yet it avoids the substantial computational cost of time-stepped simulation. In the following, we introduce two possible solutions for tackling this challenge.

5.2 Truck Dispatching Simulation Accuracy

The primary aim of a container port is to maximize the number of ships served within a given time period and thereby increase the port's turnover efficiency. As illustrated in Fig. 1.1, container ports comprise two significant areas – the berths and the yards – linked by container trucks.

Fundamentally, there are two primary operations within a container terminal: loading and unloading containers onto or from ships at the berths, and to or from the containers in the yards, with the help of container trucks. There are two prevalent container sizes: the small Twenty-Foot Equivalent Unit (TEU) and the large container, equivalent to two TEUs. Each truck can carry one large container or two small containers. Typically, a small container is bundled with another small container to form a standard `task`, while a large container is treated as a task. However, in specific scenarios, dynamic truck dispatching algorithms may be necessitated to dispatch two separate small containers when possible.

Throughout loading and unloading operations, trucks convey containers between yards and ships. QCs are tasked with loading or unloading containers from and onto ships, while YCs manage the yard operations. QCs can handle one large container or two small containers in one action, whereas YCs can only operate one small or large container due to differences in clamp types. The fact that QCs and YCs can only operate containers from one truck at a time results in waiting periods and congestion under these cranes. Moreover, as illustrated by the passage direction arrows in Fig 1.1, trucks must comply with the port's traffic regulations. Certain routes permit bidirectional traffic, whereas others allow for unidirectional movement only. This regulatory structure is critical for maintaining a safe and efficient circulation of trucks within the port.

All container operations conform to pre-designed work instructions (tasks). Each task comprises unique information such as task id, container id, source node, destination node, task type (load/unload), container size, weight, and bound task id. To maintain a balanced ship's allotment, all tasks must be executed in sequence, except for unloading tasks where the sequence can be swapped within a certain number (sn).

Considering the crucial role of maintaining continuous operation of the QCs for optimal ship operation efficiency, the QC waiting time (Quay Crane Waiting Time (QCW)) has emerged as a vital metric for evaluating dispatching strategies. Let s_{ij} and e_{ij} denote the start and end times, respectively, for task j of QC i, with n_i representing the total number of tasks for QC i and m being the number of QCs. The total QC waiting time can be represented in Equation (5.1). Another important metric is the number of TEUs processed per hour (TEU/h, TEUs Processed per Hour (TPH)) for the entire port. This metric has a direct impact on the ship docking time and overall turnover efficiency of the port. Let n be the total number of tasks, $size_i$ represent the size of task i, E denote the end times for the task set, and S represent the start times for the task set. The TPH can then be expressed in Equation (5.2).

$$QCW = \sum_{i=1}^{m} \sum_{j=2}^{n} s_{ij} - e_{i(j-1)} \tag{5.1}$$

$$TPH = \frac{\sum_{i=1}^{n} size_i}{\max E - \min S} \tag{5.2}$$

Given the formidable challenges of testing dispatching strategies in real-world scenarios, most research efforts have turned to surrogate simulators. These are used to replicate the port operation process and to train and test dispatching algorithms (He et al., 2013; Jaoua et al., 2012; Sarmiento et al., 2019). Typically, work instructions are input into these simulators, which then simulate the entire task completion process. The dispatching algorithms within the simulator distribute trucks, subsequently outputting TPH and QCW values to evaluate algorithmic performance. Although many studies have demonstrated substantial improvements in these metrics, the development of their surrogate simulators frequently neglects the interaction between trucks during transit. Although it's customary to exclude ostensibly non-essential processes when constructing surrogate simulation models, this approach can introduce complications in specific applications, such as truck dispatching in container terminals.

Our tests at Ningbo Meishan Port highlighted that dispatching algorithms, trained using conventional event-based simulators, do not consider intersection conditions during truck dispatching. This omission can lead to inadequate dispatching decisions, resulting in an uneven distribution of trucks across QCs, with some experiencing a surplus and others a shortage. Such imbalances necessitate constant monitoring of QC statuses by port dispatching operators, who must make on-the-fly adjustments to dispatching plans to maintain operational efficiency within the port. It is clear from these findings that there is a need for a more sophisticated surrogate simulator to train and test dispatching algorithms effectively, ensuring that the strategies developed are both feasible and efficient in real-world container port settings.

Furthermore, as detailed in Section 5.7, our analysis revealed that traditional event-based simulators often overestimate TPH and QCW values. Such miscalculations could lead to misjudgments of algorithm performance in real-world scenarios. Recognizing this limitation, we were motivated to develop an innovative truck dispatching simulator. By integrating a data-driven, learning-based approach, we aim to enhance simulation accuracy without significantly increasing computational overhead. In the subsequent sections, we outline the structure of our simulator and discuss the algorithms implemented to emulate truck movements at intersection nodes accurately.

5.3 Learning-Based Methods for Simulation

In our endeavor to simulate the truck movement process more precisely, we have segmented the traveling route of trucks, which typically extends from crane to crane. The route now comprises intermediary stops at intersection nodes, as depicted in Fig. 5.1. To elucidate the role of these intersection nodes in the simulation, we have drawn a logic map of the previous sample container port map as seen in Fig. 1.1.

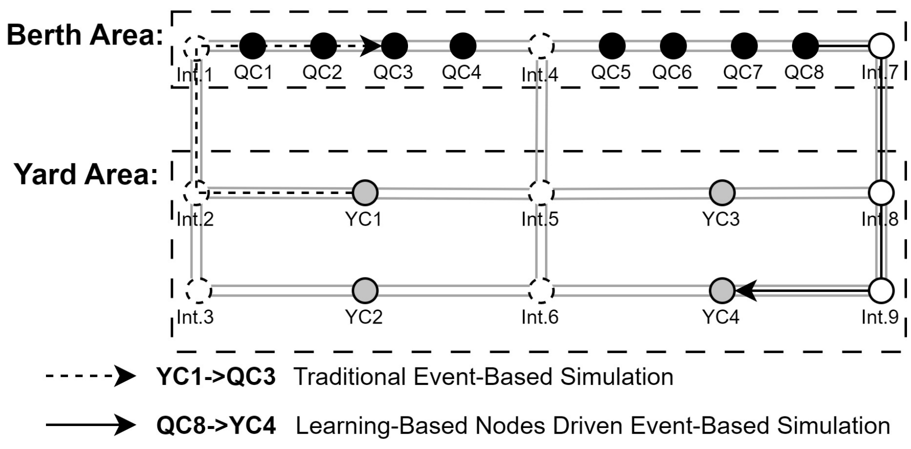

Berth Area:
Int.1
QC1
QC2
QC3
QC4
Int.4
QC5
QC6
QC7
QC8
Int.7
Yard Area:
Int.2
YC1
Int.5
YC3
Int.8
Int.3
YC2
Int.6
YC4
Int.9
YC1->QC3 Traditional Event-Based Simulation
QC8->YC4 Learning-Based Nodes Driven Event-Based Simulation

Figure 5.3: Example of Truck Routing in Logic Map

As illustrated in Fig. 5.3, the traditional event-based simulation calculates the truck movement path based on the actual path length but neglects intersections. This method is akin to a truck moving instantaneously from YC1 to QC3, bypassing all intermediate steps such as Intersection (Int.) 1 and 2. However, our proposed learning-based simulation method moves the truck from node to node. All nodes on the path, such as Int. 7, 8, and 9, are considered, as demonstrated in the figure. This segmented path can more accurately simulate truck movement, considering all actions and interactions at intersection nodes. For instance, a situation as depicted in Fig. 5.2, wherein situation 1 occurs at an intersection node along the truck's route. Upon the truck's arrival at this intersection node, the intelligent node autonomously assesses its current state. Drawing from this assessment, the node determines a passage time that accommodates for the truck's waiting, deceleration, and acceleration phases. This comprehensive computation of passage time not only enables adjustments to truck speeds but also assists in accurately simulating truck actions at intersections. As a result, it significantly enhances the precision of the truck dispatching simulation.

The calculation incorporates the following state factors in this paper:

- The current number of trucks at the node.
- The maximum passage time of the current node trucks.
- The truck number coming from the left/right/up/down.
- The closest truck distance from the left/right/up/down.

According to the above description, the truck operation function is presented in Algorithm 5. In this function, in addition to a dispatchable truck list, and a task list to be completed, *routing* is a procedure that calculates the shortest path from the start crane to the end crane of each task while respecting traffic regulations set by the port. To simplify the experimental process, the route from one crane to another is a pre-computed fixed path. For the path from QC8 to YC4, the traditional event-based simulation output merely the O-D pair [QC8, YC4] with a constant travel time, with truck interactions at intersections being overlooked. In our proposed learning-based simulation, however, the output path will be [QC8, Int.7, Int.8, Int.9, YC4]. When passing the intersection nodes, the function *n.passing*() will generate a passing time for this node by using the forecast model with the environmental states as inputs. In this way, the

Algorithm 5 Truck Operation Function

```
Require: $truck, task, routing, env$
 1: $truck.start\_task(task)$
 2: $truck.path \leftarrow routing(truck.pos, task.start\_crane)$
 3: for $n$ in $truck.path$ do
 4:     $truck.wait(truck.travel(n))$
 5:     $truck.pos \leftarrow n$
 6:     if $n.type = crane$ then
 7:         $truck.wait\_until(n.available)$
 8:         $truck.wait(task.operate())$
 9:         if $truck.pos = task.twin\_task.start\_crane$ or $truck.pos = task.twin\_task.end\_crane$ and $task.twin\_task$ then
10:             if $n.type \neq quay\_crane$ then
11:                 $wait(task.twin\_task.operate())$
12:             end if
13:         end if
14:     else if $n.type = intersection$ then
15:         $truck.wait(\mathbf{n.passing}(env.states))$
16:     end if
17:     $truck.leave(n)$
18: end for
19: $truck.end\_task(task)$
```

traffic states at each intersection node are modeled as a decisive factor in the forecasts.

Despite the considerable advances in our simulation strategy, we still face a fundamental issue: how to use state factors to reliably predict the truck passing time to boost the precision of our simulations. Conventionally, if we can access historical data that elucidates the relationship between the truck's passing time and intersection status parameters, a forecast model can be built to leverage this rule to calculate passing time. Moreover, it is possible to extract this correlational data from detailed truck GPS information to build a deeper understanding.

Container ports present a significant challenge due to a scarcity of high-quality truck GPS data. This dearth of reliable information impedes our ability to identify hidden relationships between various operational parameters. Despite substantial investments in real time kinematic (RTK) devices and numerous engineering efforts, the GPS-unfriendly port environment still yields a high proportion of erroneous positioning data. In the absence of precise GPS data for trucks and information on the queuing wait time at each crane, the accurate replication of the trucks' historical route poses a substantial challenge.

To address this issue, we propose a learning-based method that estimates intersection passing times using a reward-based system. This approach facilitates the discovery of traffic flow rules, even in contexts where operational data might be sparse or limited. Learning-based methods like GP and RL do not rely on explicit programming or hand-crafted rules. Instead, they learn from available data, iterating over numerous cycles, making and learning from errors, and progressively improving their predictions. The central mechanism is a reward-based system: the more a prediction or model's action aligns with actual outcomes, the higher the reward. The algorithms strive to maximize these rewards, thereby gradually improving their performance, even in the face of limited data.

GP and RL are particularly suited to the task at hand due to their flexibility, their ability to manage complex, non-linear relationships, and their resilience against noisy or sparse data. These attributes make them ideal for enhancing the accuracy of our truck dispatching simulator.

In the subsequent sections, we provide detailed insights into the selected algorithms and learning methods, highlighting their contribution to accurately predicting intersection crossing times despite data limitations.

5.4 Data-driven Genetic Programming in Simulation

GP is an evolutionary computation method inspired by the principles of biological evolution. Its essence lies in refining a population of solutions through consistent modifications of crossover and mutations. Hence, by designing a fitness function that encourages the generated individuals to provide truck intersection passing times that match the historical data best, we can allow the model to learn hidden rules through its evolution.

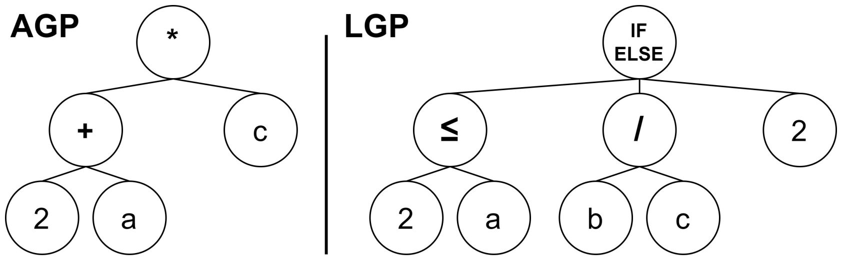


Figure 5.4: AGP and LGP Structure

GP employs various representations, but the tree structure is the most common and easy to understand and is also adopted in this study. As depicted in Fig. 5.4, GP can be divided into two types: arithmetic genetic programming and logical genetic programming, contingent upon whether logical operators are included or not.

In this paper, AGP utilizes operations such as addition, subtraction, multiplication, and protected division. Meanwhile, LGP integrates additional logic operators, including greater than or equal to, less than or equal to, if-else, and, or, maximum, and minimum. The terminals in GP are the state factors enumerated in the previous subsection, amounting to 10 state factors plus one random constant.

In the context of the learning-based simulation, GP individuals act as the *n.passing*() function to compute the passing time of trucks at intersections. To train these GP individuals, the fitness function is set up according to Equation (5.3), which combines the metrics detailed in Equations (5.1) and (5.2).

In the equation above, S/s and E/e represent the task start and end times observed in actual operations, respectively, analyzed collectively and individually. Correspondingly, S'/s' and E'/e' symbolize the same times as the simulated results.

The fitness function, denoted as R_f, measures the disparity between the actual data from operational records and the simulation outcomes. This function focuses on two main metrics: TPH and QCW, as observed in the training dataset. By gauging the performance of GP individuals, this fitness function aids in pinpointing the entities possessing accurate knowledge necessary for estimating truck intersection passing times. As a result, the simulation developed is highly reflective of actual operational data. The parameter δ is implemented to balance the relative importance of the two metrics, and for the purpose of this study, it is set to 1. It's important to note that in this paper, we don't distinguish between different intersections; we instead employ a single model to fit all intersections. This approach is consistent across both GP and RL methods.

$$\begin{aligned} R_f = \delta \sum_{i=1}^{m} \sum_{j=2}^{n} |s_{ij} - e_{i(j-1)} - s'_{ij} + e'_{i(j-1)}| \\ +|\max E - \min S - \max E' + \max S'| \end{aligned} \tag{5.3}$$

The GP evolution process, as outlined in Algorithm 6, continually uses genetic operations such as crossover and mutation methods to produce new individuals. This method screens out and eliminates individuals with low fitness, allowing the entire population to evolve toward individuals with higher fitness. This paper employs tournament selection for better control on the selection pressure. The crossover and mutation methods align with those mentioned in our previous work (Chen et al., 2022). Notably, since the fitness of GP signifies the disparity between metric values of real data and simulated results, lower fitness indicates higher simulation accuracy.

Algorithm 6 AGP, LGP Evolution Function

```
Require: Initial Parameters $initial$
  $p \leftarrow NewPopulation$
  $p.initial\_individuals(initial.population\_size)$
  $generation \leftarrow 0$
  while $generation < initial.max\_generation$ do
      $p.calculate\_fitness()$
      $p.penalize\_long\_individuals()$
      $next\_generation \leftarrow NewPopulation$
      while $next\_generation.size() < p.size()$ do
          Insert an $individual$ to $next\_generation$ by
          Crossover, Mutation, or Reproduction in $p$
      end while
      $p \leftarrow next\_generation$
      $generation \leftarrow generation + 1$
  end while
```

While our experimental results indicate that both AGP and LGP can significantly improve the accuracy of simulations, they have a key limitation: the fitness function of traditional GP can only be calculated upon completion of the entire simulation process. This delay means that the valuable data generated during the simulation cannot be fully exploited to optimize the accuracy of simulations in real-time.

In light of these considerations, we believe that leveraging RL to learn the truck's passing time at intersections could greatly enhance the efficiency and accuracy of our simulations. By using RL, we can fully utilize the historical data at our disposal and acquire more precise and informative insights in real-time, which ultimately leads to more accurate simulations.

This RL-based approach will be discussed in detail in the following subsection.

5.5 Reinforcement Learning in Simulation

RL is a machine learning algorithm that trains an agent to select an appropriate action to obtain maximum rewards. Unlike GP, which relies on evolutionary principles to refine its predictive models, RL leverages its ongoing interaction history, often via mechanisms like experience replay, to progressively enhance its decision-making policies. In the traditional GP context, R_f is calculated as a reward (fitness) after completing all task simulations, providing limited feedback to the learning process. In this environment, the trajectory-based experiences gathered by the agent do not offer substantial information about the quality of the action, which in turn considerably impacts the quality of the final simulation results.

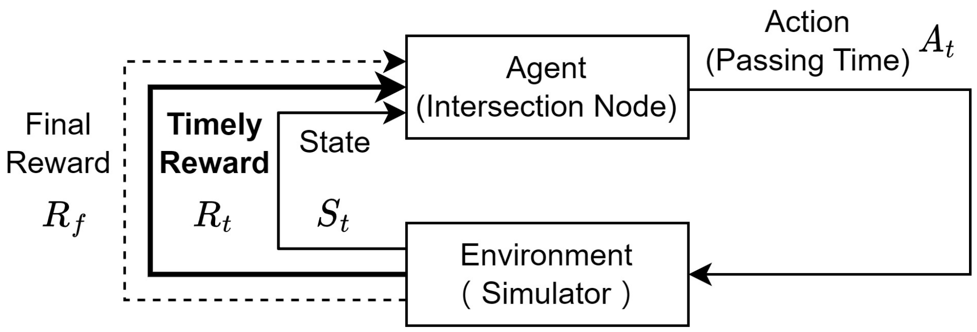


Figure 5.5: RL Structure

However, RL introduces a notable paradigm shift in the learning approach compared to GP. Rather than being completely disconnected from the environment during the evolutionary process, as is the case with GP, RL allows the agent to engage in continual interaction with the environment. The agent performs actions and receives feedback in the form of rewards, as depicted in Fig. 5.5. This dynamic interaction facilitates a more adaptive learning process and can help to optimize the actions more effectively. Specific aspects of this feedback mechanism, including its immediacy and influence on the learning process, will be explored in detail in the results section.

Thus, we introduce a timely reward, R_t, formulated as per Equation (5.4) in the RL method. The symbols s_ij and e_ij represent the start and end times in the actual data for the jth task of the ith QC, respectively, while s'_ij and e'_ij denote the start and end times in the simulation. The reward R_t is calculated as the discrepancy between the actual and simulated truck movement times for each task. As with the reward R_f, smaller R_t values signify a higher simulation accuracy.

$$R_t = |E_{ij} - S_{ij} - E'_{ij} + S'_{ij}| \tag{5.4}$$

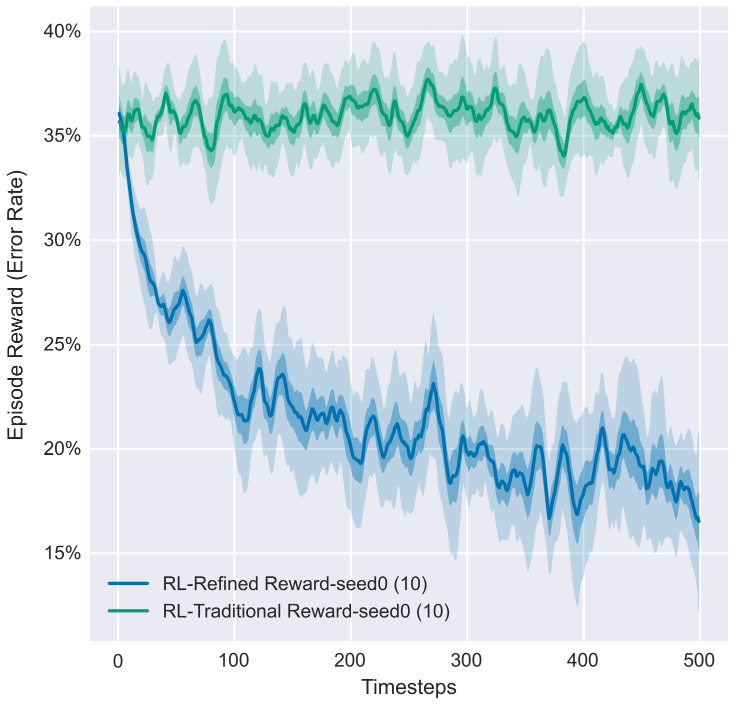


Figure 5.6: Performance of RL with Different Rewards

The novel aspect of our RL approach is the concurrent use of the real-time computed reward R_t and the episode-end reward R_f to guide the learning process. This combined guidance allows the RL agent to learn the logic of truck operations at the intersection more effectively from historical data. By incorporating the real-time computed reward R_t following each action A_t by the RL agent, we can better use the previously ignored data on the truck movement time for each task. Consequently, the agent can acquire more information, improving the quality of actions and, ultimately, a more accurate simulation.

In our experiment on the test dataset, we ran 10 iterations and trained for 500 episodes to assess the impact of different rewards on RL performance. As illustrated in Fig. 5.6, using the traditional final reward approach did not lead to effective convergence. RL failed to acquire helpful knowledge, resulting in a high simulation error rate. Conversely, our proposed combination of two types of rewards significantly improved RL's convergence. This approach enabled RL to converge on complex problems, overcoming challenges associated with RL in intersection simulations and substantially increasing the accuracy of the simulations.

In the context of this research, RL is expected to output the truck crossing time at an intersection as a continuous variable. This requirement presents a challenge to traditional RL methods such as deep Q-learning (Mnih et al., 2013) (DQN) and double deep Q-learning (Van Hasselt et al., 2016) (DDQN), which are not optimized for handling continuous action spaces effectively. To resolve this issue, we adopt the proximal policy optimization (PPO) method to calculate the truck crossing time at the intersection.

The RL training environment used in this study is similar to the one for the port dispatch simulator previously discussed. Like in the GP method, the RL agent performs the function $n.passing()$ in Algorithm 5, producing a continuous number representing the time a truck took to cross the intersection, given the current environment parameters. The environment state is defined by the 10 state parameters as described in the below Sections.

During the training process, the simulator provides the agent with intersection state information S_t, and the agent outputs the predicted intersection passage time based on its learned knowledge. Each time the agent makes an action A_t, the simulator returns a reward R_t. Once all actions are completed, a final reward R_f is computed. The entire training process of the PPO-based RL algorithm is illustrated in Algorithm 3.

Experimental results indicate that RL demonstrates superior simulation accuracy compared to GP. This improvement is largely attributed to the inclusion of the timely reward R_t, which provides RL agents with an immediate response to their actions, leading to more refined decisions. However, this added precision has a significant drawback: increased computational cost. Our experiments show that utilizing RL for simulation requires approximately twice the computational time of using GP.

The higher computational cost results from the RL-generated agent being invoked to predict every time a truck passes an intersection. While individual predictions may take minimal time, a complete simulation requires hundreds or thousands of computations. This cumulative computation time can render the RL's learning-based simulation too time-intensive when used to train dispatching strategies, thereby limiting its practicality.

To strike a balance between accuracy and computational efficiency, we propose a hybrid GP and RL simulation approach. This strategy involves using GP to simulate less important intersections and RL to simulate intersections of high importance. By leveraging the strengths of both methods in this way, we aim to minimize the computational cost while preserving the accuracy of the simulation. The effectiveness and efficiency of this GPRL-H simulation approach will be examined in the subsequent part.

5.6 Hybridizing GP and RL in Simulation

Unlike GP, wherein the acquired knowledge is directly translatable into arithmetic functions, thereby substantially accelerating each truck pass time calculation, RL enhances simulation accuracy and considerably extends the simulator's runtime. Empirical results indicate that RL-based simulations demand about thrice the computational time compared to GPs. To reduce this simulation time while preserving the accuracy of the simulations, this paper puts forward an innovative approach: the GPRL-H method for fast estimation of truck passage times at intersections.

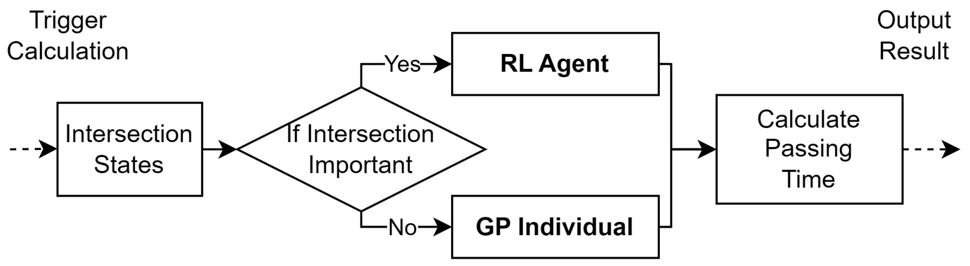


Figure 5.7: GPRL-H Method Flow Chart

As visualized in Fig. 5.7, when the simulation necessitates the computation of intersection passing time, the GPRL-H method determines whether to employ GP or RL for this calculation based on the current intersection's significance where the truck is positioned. Our analysis of historical truck movement data revealed that certain intersections, particularly those at critical locations, frequently witness multiple trucks crossing simultaneously, while at other intersections, such instances are rather rare. Consequently, we hypothesize that these critical intersections, given their complex traffic conditions, demand a detailed simulation facilitated by RL. In contrast, intersections with less complex traffic patterns could be effectively simulated using GP, yielding a level of accuracy similar to that of a comprehensive RL simulation. This hypothesis forms the basis for our proposal of the GPRL-H method, striking a balance between simulation accuracy and computational efficiency.

Nevertheless, the GPRL-H approach brings forth a challenge - determining the importance of each intersection. Given the unavailability of accurate GPS data, direct analysis of each intersection's congestion levels using historical GPS data to determine intersection significance is unfeasible. Therefore, drawing inspiration from the Pareto chart (Harvey and Sotardi, 2018), we propose a data-driven method incorporating the previously mentioned RL-based intersection simulation to evaluate the significance of various intersection nodes.

Initially, let's denote that there are a total of o intersections within the port. In this research, we have $o = 66$, and all intersections in the default simulator sd are controlled by RL agents. The process can be outlined as follows:

1. Train the GP-based and RL-based agents independently on the training sets.
2. Replace the RL agent of one of the o intersections in the sd with the GP individual sequentially, generating o distinctive GPRL-H simulators.
3. Execute these substituted o simulators on test data sets and tally the final simulation results.
4. Implement the paired sample t-test method to compare the results of

the o simulations with the sd, and compute the t-value. The higher t-test results indicate a greater discrepancy between the outcomes before and after removing a specific node. This suggests the higher importance of the node, implying that it should not be substituted with GP for control.

5. Choose the replaced intersection in the simulator with the lowest t-value (lowest influence on simulation accuracy) as the least important intersection and replace the RL agent at this intersection with a GP individual in default simulator sd. Set $o = o - 1$ to save simulation time and return to step 2 to continue replacing the next intersection until GP individuals control all intersections.

Moreover, we utilize the LGP-generated individual as the GP part in the GPRL-H method. This approach has been adopted due to its superior performance compared to AGP in our experiments while maintaining a comparable execution time. Following the previously outlined steps, we derive the intersection importance diagram in Fig. 5.8. This diagram shows that when the performance of RL agents at all intersections is considered 100%, replacing 40 out of 66 intersections with GP-regulated agents retains approximately 96% of the performance characteristics seen in a simulation entirely driven by RL agents. As a result, the proposed GPRL-H method in this paper defaults to controlling these 40 less significant intersections using GP, thereby ensuring an effective balance between simulation accuracy and speed.

Our experimental results reveal that the proposed GPRL-H method combines the strengths of both GP and RL and strikes an effective balance between simulation accuracy and computational cost. We propose that this innovative method can be applied not only to the port dispatch sim-

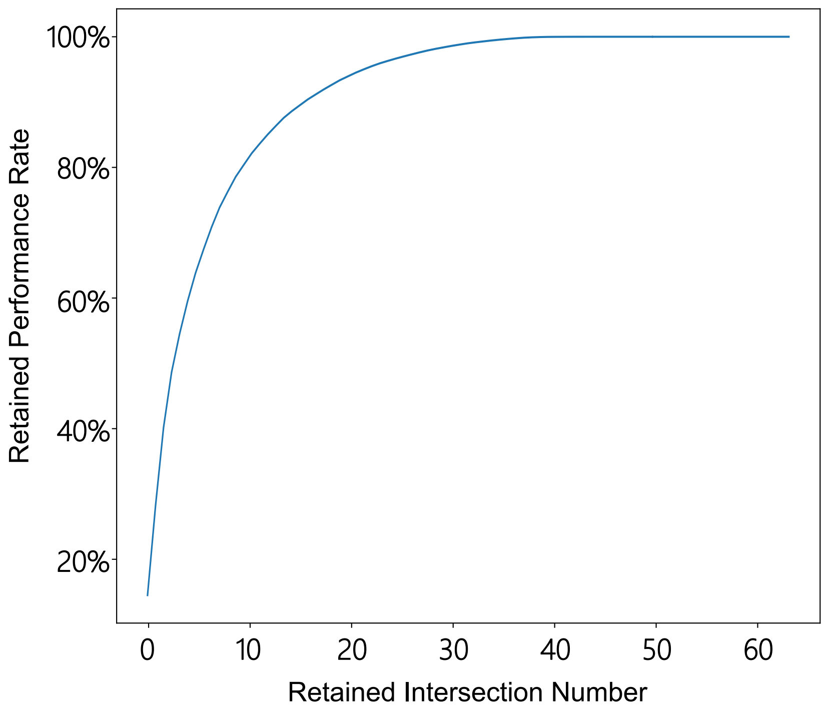


Figure 5.8: Intersection Importance Analysis Result

ulation discussed in this paper but also to a broader range of road traffic simulation applications, including factory-automated guided vehicle (Automated Guided Vehicle (AGV)) path simulation, urban traffic simulation, and mine vehicle scheduling. To demonstrate the superiority of the GPRL-H algorithm, a series of experiments and comparisons will be conducted in the next section.

5.7 Experiments and Discussion

In this section, we present the validation of the traditional event-based simulation and the learning-based AGP, LGP, RL, and GPRL-H methods that we have proposed, utilizing real-life historical data obtained from the Ningbo Meishan Port. Our primary objective is to highlight the excellent performance of our emphasized GPRL-H method. As articulated in Section

5.2, our central focus remains on the computation time and errors of two principal metrics: QCW and TPH.

The validation leverages real-world data and maps from Ningbo Meishan Port. The designated truck routes comply with the stipulations of the port; each route from crane to crane is unique and pre-set. The historical operational data chiefly constitutes the initiation s_{ij} and termination e_{ij} timings for the truck operations at the loading crane i and task j, supplemented with the start and end times of operations at the unloading cranes.

We have processed 20 days of historical operational data to generate 10 training and 10 test sets. Each set comprises roughly 20,000 job tasks, approximating the daily job volume at the port. The port comprises 5 berths, 35 QCs, and 75 YCs. Moreover, there are 66 intersections, and the number of trucks varies between 100 and 200, which is determined by the actual count of trucks in the historical data.

The parameter configurations and evolution methodologies for all GP methods are consistent with our prior work on AGP and LGP methods (Chen et al., 2022), having a population size of 1024, and crossover, mutation, and reproduction rates of 60%, 30%, and 10%, respectively. The GPRL-H method utilizes RL to regulate the 26 crucial intersections and employs LGP to fit the remaining 40 less critical intersections. The AGP, LGP, RL, and GPRL-H methods are trained on the 10 training sets using 100 distinct random seeds for 1000 generations each.

For the RL component of our study, we have chosen PPO and configured it to use two distinct deep neural networks. One of these networks serves as the policy network, or the 'actor network,' while the other functions as the value network, often termed the 'critic network.' The learning rate has been predetermined at 0.0003 for the actor network, and for the critic network, it is set at 0.001. These neural networks consist of layers with neuron quantities as follows: 10, 100, 180, and 1, respectively. The Rectified Linear Unit (ReLU) is implemented as the activation function across all layers. In the experiment, the error percentage for TPH and QCW is calculated using the formula $Error = \left(\frac{\text{Simulated TPH or QCW}-\text{Historical TPH or QCW}}{\text{Historical TPH or QCW}}\right) \times 100$. The final performance of the models, post-training, on the training sets is illustrated in Table 5.1.

Table 5.1: Traditional, AGP, LGP, RL, GPRL-H Methods Training Results

No.	Historical		Traditional		AGP		LGP		RL[1]		GPRL-H[1]	
	TPH	QCW	TPH	QCW	TPH	QCW	TPH	QCW	TPH	QCW	TPH	QCW
1	472.04	70.94	719.22	90.43	624.20	41.05	409.13	83.29	476.21	70.12	466.90	71.95
2	433.40	78.52	548.19	71.09	431.56	78.89	601.79	45.45	426.79	79.82	444.74	76.30
3	393.73	86.32	585.19	107.41	489.23	67.56	387.75	87.49	323.06	100.19	327.53	99.32
4	446.71	75.91	613.20	114.21	481.70	69.04	432.70	78.66	532.21	59.12	538.52	57.88
5	431.88	78.82	403.06	130.02	380.11	88.99	463.59	72.60	501.68	65.12	503.52	64.75
6	371.38	90.71	390.09	148.52	381.82	88.65	269.85	110.64	368.59	91.25	386.79	87.68
7	436.93	77.83	621.52	92.81	267.84	111.04	374.50	90.09	429.31	79.33	435.45	78.12
8	470.64	71.21	581.70	76.30	648.65	36.25	544.00	56.80	476.61	70.04	476.90	69.98
9	371.41	90.70	345.27	133.50	531.44	59.27	329.86	98.86	371.47	90.69	376.88	89.62
10	382.13	88.59	659.96	97.29	404.18	84.26	504.38	64.58	380.01	89.01	394.86	86.09
Avg.	421.02	80.96	546.74	106.16	464.07	72.50	431.75	78.85	428.59	79.47	435.21	78.17
Error	**N.A.**	**N.A.**	**36.43%**	**34.32%**	**24.36%**	**26.25%**	**17.89%**	**18.22%**	**6.17%**	**6.50%**	**6.86%**	**7.07%**

[1] RL and GPRL-H are significantly different from other methods, $p < 0.05$.

As outlined in the table, the unit of measure for TPH is TEU/hour, while that for QCW is hours. The error term represents the average absolute deviation in comparison with historical data across the ten datasets. Notably, our learning-based simulations consistently outperform conventional event-based simulations, irrespective of the intersection simulation method employed. We also incorporated an error parameter into our experiments to gauge the accuracy of various simulation techniques. It is crucial to clarify that this error is not merely an average of the last two metrics but rather represents the cumulative error in predicting individual segments of truck travel time. Hence, scenarios may arise where the final metrics exhibit a smaller mean error but a larger cumulative error. Traditional event-based simulations display a significant error rate of approximately 35%. In contrast, AGP and LGP demonstrate considerably improved performance, with respective error rates of around 25% and 18%. Intriguingly, the incorporation of logical operators into GP enables LGP to outperform AGP. This suggests that intersection simulation is not a simple linear problem; it encapsulates diverse conditions that warrant the inclusion of logical operators for more accurate fitting. Remarkably, reinforcement learning (RL) exhibits a significantly lower error rate of around 6.5%, substantiating our claims in Section 5.5 that GP's performance is considerably influenced by its singular interactions with the environment during each evolutionary cycle. Although LGP manages some less critical nodes, the GPRL-H method maintains performance metrics comparable to RL, with an error rate of approximately 7%, essentially on par with RL.

Subsequently, we deployed the AGP, LGP, RL, and GPRL-H methods, initially trained on the training sets, into the test sets as shown in Table 5.2. To circumvent potential distortion due to disparities in the historical data when comparing the training and test sets, we elected test sets closely aligned with the training sets and historical data. The performance of the conventional method remained relatively unchanged, with an error rate persisting around 36%. Meanwhile, the AGP, LGP, RL, and GPRL-H methods all experienced nominal decreases, though these were not significant, merely around 1%-2%. This suggests our training did not lead to overfitting, and the intersection traffic rules gleaned from the learning-based methods are indeed generalizable, yielding effective performance even on datasets not previously encountered. This further affirms the versatility of our proposed method and its potential applicability in other simulations.

Table 5.2: Traditional, AGP, LGP, RL, GPRL-H Methods Test Results

No.	**Historical**		**Traditional**		**AGP**		**LGP**		**RL**[1]		**GPRL-H**[1]	
	TPH	QCW	TPH	QCW	TPH	QCW	TPH	QCW	TPH	QCW	TPH	QCW
1	457.65	73.76	740.50	18.21	614.41	42.98	713.78	23.46	433.02	78.60	434.44	78.32
2	344.21	96.04	413.43	82.45	354.14	94.09	294.82	105.74	306.60	103.43	302.29	104.27
3	482.55	68.87	553.73	54.89	569.48	51.80	463.86	72.54	516.05	62.29	538.12	57.96
4	362.13	92.52	520.68	61.38	273.62	109.90	247.04	115.12	327.51	99.32	329.73	98.88
5	492.18	66.98	614.07	43.04	480.49	69.28	413.87	82.36	504.43	64.58	509.66	63.55
6	405.43	84.02	461.49	73.01	337.86	97.29	430.48	79.10	455.10	74.26	468.66	71.60
7	418.06	81.54	544.09	56.79	303.39	104.06	367.91	91.39	410.01	83.12	408.67	83.38
8	433.57	78.49	504.59	64.54	500.73	65.30	318.17	101.15	403.09	84.48	419.80	81.19
9	457.73	73.75	720.02	22.24	564.02	52.87	459.40	73.42	412.24	82.68	407.73	83.57
10	365.82	91.80	524.78	60.58	113.09	141.43	377.77	89.45	332.69	98.30	332.34	98.37
Avg.	421.93	80.78	559.74	53.71	411.12	82.90	408.71	83.37	410.07	83.11	415.14	82.11
Error	**N.A.**	**N.A.**	**35.65%**	**37.46%**	**26.13%**	**26.10%**	**18.23%**	**19.29%**	**7.74%**	**7.71%**	**8.46%**	**8.58%**

[1] RL and GPRL-H significantly differ from other methods, $p < 0.05$.

Additionally, we computed the simulation time (SimT) consumed over each instance for each method. As presented in Table 5.3, it is evident that the standard event-based simulation is the most time-efficient, with an average simulation duration of approximately 5 seconds. The mean simulation times for AGP and LGP do not exhibit a notable difference, both hovering around 11 seconds, which is approximately twice as long as that of the standard method. However, RL incurs the longest simulation time, with an astounding average of 35 seconds—sevenfold that of the standard method. Despite RL exhibiting the most impressive performance among all methods, such an extensive runtime is untenable. In the process of employing the simulator to train port container truck dispatching strategies, many simulation executions are required. A simulation time seven times longer implies a corresponding increase in training duration when using the standard simulator, substantially impeding training efficiency. In contrast, our innovative GPRL-H method necessitates an average simulation time of around 17 seconds, merely half of that required by the RL method, while retaining near-equivalent simulation precision. This underscores that the GPRL-H method achieves a commendable equilibrium between two critical metrics: simulation time and simulation accuracy. It is capable of substantially reducing simulation time while preserving adequate simulation precision.

The empirical results compellingly demonstrate the efficacy of our learning-based simulation methodology for container port truck dispatching. Under the purview of our proposed framework, applying AGP, LGP, RL, and GPRL-H methodologies to model truck throughput times at intersections substantially enhances the simulation's precision. Our GPRL-H method warrants specific mention. By amalgamating the computational speed of GP and the superior accuracy of RL, this approach is uniquely positioned

Table 5.3: Traditional, AGP, LGP, RL, GPRL-H Methods Simulation Time (second)

Set	**No.**	**Tradit.**	**AGP**	**LGP**	**RL**	**GPRL-H**
Train	1	4.88	9.42	11.73	36.55	17.70
	2	5.36	11.33	8.93	28.25	17.04
	3	4.91	11.87	11.73	34.74	16.13
	4	5.04	10.64	10.92	36.82	16.29
	5	4.21	10.36	12.55	34.35	17.20
	6	4.20	11.28	11.43	39.82	17.70
	7	4.46	10.01	10.94	37.54	16.06
	8	5.28	13.10	10.66	34.34	16.79
	9	5.33	11.16	11.24	40.81	20.76
	10	5.88	10.52	11.41	35.74	17.59
Test	1	4.76	10.79	9.73	35.11	18.22
	2	5.10	11.16	10.50	31.90	17.58
	3	5.98	10.57	11.63	36.85	18.22
	4	5.31	10.34	11.89	44.54	16.04
	5	4.82	9.48	11.38	41.09	15.49
	6	4.58	11.93	12.26	35.37	18.55
	7	5.22	10.80	13.16	31.54	17.62
	8	4.57	11.66	10.84	37.00	18.62
	9	4.52	9.03	11.26	32.41	17.06
	10	4.93	12.00	11.19	45.08	19.29
	Avg.	**4.97**	**10.87**	**11.27**	**36.49**	**17.50**

to deliver high-grade simulation accuracy within a significantly reduced time frame.

This study's key strength resides in its capacity to provide a sophisticated, learning-based tool that transcends conventional event-based simulations. The GPRL-H method, by being anchored in both GP and RL, exploits the strengths of these paradigms to yield a potent framework that is both time-efficient and high in precision. This innovative fusion of techniques renders the model responsive and adaptable, making it a powerful tool for real-world applications.

The empirical data derived from our work suggests a promising degree of generalizability for our learning-based GPRL-H simulation method. Our results support the claim that this method not only provides enhanced performance in port dispatching problems but also carries the potential for applicability in similar contexts. However, we acknowledge that a broader applicability claim requires further validation across multiple scenarios and domains.

The integration of GP and RL in our method is an innovative approach that could be employed in other fixed-area vehicle simulations, to improve simulation accuracy. While surrogate models are prevalent in simulation studies, our unique combination of GP and RL provides a novel contribution to the field. We must note, however, that the true reach of this method and its applicability beyond port dispatch is a promising prospect that warrants further investigation and corroboration. Its potential to significantly contribute to sectors where precise and efficient simulation is paramount is a compelling avenue for future exploration.

5.7.1 Discussion and Supplementary Experiments

In the previous experimental section, we provided evidence supporting the efficacy of our proposed learning-based simulation approach. In this section, we delve further into the necessity of implementing intersection simulations and the importance of integrating learning-based methods for more precise simulation. The cornerstone of our hypothesis is that the waiting times incurred by trucks at intersections during container transport markedly influence the overall transportation time. By enhancing the precision of truck passage time simulations at intersections, we can significantly improve the overall accuracy of the container port truck dispatch simulation. In turn, refining the simulation of truck passage through intersections bolsters the accuracy of the entire simulation. Moreover, the process of calculating waiting times for trucks at intersections presents a complex task, necessitating the consideration of various conditions present at the intersection during the truck's passage. Thus, a constant intersection waiting time falls short in accurately simulating varying truck traffic conditions.

Upon reviewing our experimental data, we noted that the standard event-based simulation generally overestimated TPH and underestimated QWT. This observation could potentially stem from the disregard of additional transit times that trucks incur at intersections during the transportation process. Consequently, we first introduced the fixed extra travel time (Fixed Extra Travel Time (FETT)) method, which is rooted in an exhaustive search approach (Nievergelt, 2000). This method endeavors to identify a constant additional truck travel time that minimizes the average error on the training set by exhaustive searching.

As illustrated in Table 5.4, introducing a fixed extra travel time decreases the error on the training set to approximately 28%, substantiating the

Table 5.4: FETT, FIPT, DT, DNN and XGBoost's Performance and Simulation Time

Method		**Train**			**Test**		
		Average	Error	SimT	Average	Error	SimT
FETT	**TPH**	466.15	27.35%	5.37	424.34	34.50%	5.62
	QCW	72.09	29.28%		80.30	34.93%	
FIPT	**TPH**	421.48	20.26%	8.72	427.28	28.63%	8.66
	QCW	80.87	21.35%		78.73	28.95%	
DT	**TPH**	412.15	22.35%	6.35	424.34	33.50%	6.21
	QCW	82.09	23.28%		79.43	32.93%	
DNN	**TPH**	431.48	18.26%	6.87	415.28	30.63%	6.52
	QCW	77.87	17.35%		85.73	31.95%	
XGBoost	**TPH**	419.15	5.54%	7.32	432.34	27.63%	7.24
	QCW	79.89	5.26%		78.30	26.95%	

effectiveness of considering additional travel time. Nonetheless, this reduction in error is notably limited. This observation suggests that given the substantial variance in the time required for each truck transit, it is impractical to identify a universal extra truck travel time that significantly reduces the average simulation error across multiple training sets. Furthermore, it was noted that on the test sets, the error reduction achieved with the FETT method closely mirrored that of the conventional event-based method. This suggests that, while it may be feasible to identify a relatively optimal extra truck travel time for a specific subset of data sets, this value lacks broad generalizability. When the data set is changed, the error rate escalates dramatically, underscoring the necessity for more adaptable, learning-based approaches.

To further our exploration, we proposed the fixed intersection passing time (Fixed Intersection Passing Time (FIPT)) method, anchored in the intersection simulation framework elaborated in this paper. Like the FETT method, the FIPT approach also leverages an exhaustive search but instead aims to identify a fixed intersection passing time that minimizes the average error on the training set. Remarkably, this method demonstrates a significantly lower error rate on the test sets compared to the FETT method, with an error rate of approximately 20%. This figure is comparable to the error rate associated with the GP methods and is even slightly superior to that of the AGP. These findings suggest that upon the introduction of data-driven methodologies to learn intersection passing times, a relatively satisfactory solution can be achieved, even with the utilization of a simple exhaustive search method. This highlights the significance of simulating intersections and employing data-driven approaches. Through continuous testing on historical data, the exhaustive search method yielded a relatively fine solution, thereby fully substantiating the efficacy of the data-driven, learning-based simulation approach proposed in this study. Conversely, the FIPT method also exhibited a considerable increase in error on the training sets. This indicates that a fixed intersection passing time is not a universal rule; it solely represents the knowledge derived from a specific data set and does not constitute an abstract principle. This underlines the importance of incorporating sophisticated algorithms like GP and RL to learn the real-world dynamics of intersection passage.

After evaluating the impact of fixed extra travel time and fixed intersection passing time, we extended our comparative analysis to include three state-of-the-art supervised learning methods: decision tree (Decision Tree (DT)) (Myles et al., 2004), deep neural network (Deep Neural Network (DNN)) (Montavon et al., 2018), and extreme gradient boosting (XGBoost) (Chen and Guestrin, 2016). These methods were applied using the default settings in the sklearn framework (Pedregosa et al., 2011), with hyper-parameter tuning performed through a 5-fold halving grid search cross-validation method during training.

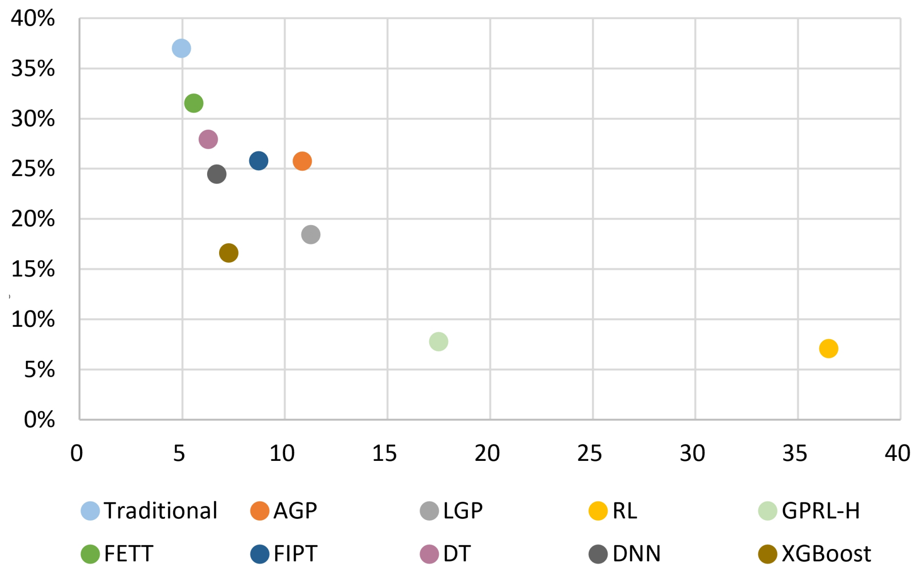


Figure 5.9: Error and Simulation Time (seconds) of All Methods in Test

While we lack real-time data on truck intersection passing, we do possess comprehensive data on total travel time between work cranes. Consequently, we employed these supervised learning algorithms to predict truck travel time under various states. All algorithms were trained on a uniform dataset, utilizing state descriptors such as task type, container type, total number of trucks operating under the current bridge crane, port-bound truck count, target crane workload, historical average travel time, and overall port truck count. The algorithms were trained using default hyperparameters from the scikit-learn library, and their performance is tabulated in Table 5.4. Both DT and DNN exhibited poor convergence on the training set with approximately 20% error, while XGBoost outperformed them with an error rate of roughly 5.5%. However, when transitioned to the test set, all three algorithms displayed an error rate of about 30%, comparable to the simple FETT and FIPT methods. Moreover, XGBoost demonstrated significant performance degradation, indicative of overfitting during training.

From these experiments, it becomes evident that predicting truck travel time solely based on current port operation states is inherently flawed, especially in the absence of data on truck intersection passages. Despite XGBoost's initially promising training performance, it too falters on the test set due to overfitting. This underscores the crucial role of incorporating intersection simulations for a more accurate representation of port truck operations. Notably, even a rudimentary method like FIPT approximates the performance of these supervised algorithms, attesting to the importance of intersection simulation in this context. Given the unavailability of intersection-specific data, employing unsupervised learning approaches in port truck dispatching simulation becomes indispensable, thereby justifying our initial focus on data-driven unsupervised learning in intersection simulation for more precise port simulation.

In the subsequent analysis, we plotted the simulation time and associated error for the Traditional, AGP, LGP, RL, GPRL-H, FETT, FIPT, DT, DNN, and XGBoost methods in Fig. 5.9. It is discernible from the graph that RL and GPRL-H methods markedly outperform the other approaches in terms of simulation error. Simultaneously, it is evident that each method holds its unique advantages when optimizing the two key objectives - simulation time and accuracy. Thus, in practical applications, decision-makers can tailor their method to meet their specific requirements. Furthermore, the graph underlines that the GPRL-H method attains a high degree of simulation accuracy without significantly increasing simulation time. This reinforces the superiority of the method introduced in this study, lending further credence to its effectiveness in balancing simulation accuracy and efficiency.

As a result of comprehensive experimental analysis, our research validates the initial hypothesis that accurately predicting the truck transit times at intersections plays a crucial role in the accuracy of simulation models. Furthermore, incorporating unsupervised learning techniques to model these transit times significantly enhances the precision of the simulation. The key findings of our study are summarized as follows:

- Experimental results substantiate the pivotal impact that truck transit times at intersections exert on the accuracy of simulation models.
- The inclusion of simple transit events in the simulation substantially improves model accuracy, emphasizing the necessity of accurately representing complex real-world conditions.
- Findings from using the FIPT method reveal the inherent complexity of intersection passing rules, which simplistic temporal metrics cannot capture.
- Traditional supervised learning methods, such as DT, DNN, and XGBoost, are inadequate for the precise prediction of truck travel times in the absence of intersection considerations.
- The observed complexities validate the necessity for the employment of unsupervised algorithms like GP and RL to decode intricate intersection rules. This lends credence to the efficacy of a data-driven, learning-based approach for addressing the challenges in container port truck dispatch scenarios.

5.8 Summary

The role of intersection simulation in container port logistics is paramount, critically shaping the precision of the final simulation outcomes. This research underscores that harnessing data-driven, learning-based simulations brings about substantial advancements in the fidelity of intersection simulations, amplifying the operational efficiency of truck dispatch processes. Given the intricate nature of intersection passing rules and their nuanced impact on logistics, the application of cutting-edge machine learning algorithms emerges as imperative for generating authentic and actionable simulations.

Our analytical endeavors, encompassing a spectrum of learning-based techniques like AGP, LGP, RL, GPRL-H, DT, DNN, and XGBoost, elucidate the distinct prowess of the GPRL-H approach. By adeptly leveraging the inherent problem structures, this methodology harmonizes the merits of RL and GP. The outcome is a commendable synthesis of high simulation accuracy and computational thriftiness, positioning GPRL-H as a premier choice for confronting multifaceted intersection simulation challenges.

This investigation not only enriches the existing literature but also marks a pivotal orientation in research ethos. It accentuates the latent potential of data-driven and learning-centric methodologies in fine-tuning transportation simulation accuracy, extending well beyond the confines of port logistics. The formidable versatility exhibited by the GPRL-H approach in our empirical exercises forecasts its expansive utility across diverse vehicular and equipment movement simulations. In our forward-looking endeavors, we envision harnessing RL as a guiding force for the evolution of GP, intending to engender more streamlined equation models for intersection simulations. This aligns with our aim of achieving swifter and more accurate logistics simulations.

Furthermore, the incorporation of nuanced transit events unequivocally elevates the authenticity of the simulations. The empirical insights derived from the FIPT method shine a spotlight on the intricate dynamics underpinning intersection passing rules. A rudimentary time metric for intersection passage proves insufficient in encapsulating these dynamics. Even if such a metric exhibits utility for a select data set, its efficacy wanes across broader datasets owing to intrinsic temporal variances. This accentuates the indispensability of sophisticated computational tools, particularly GP and RL, in deciphering the labyrinth of intersection rules. In culmination, our study robustly advocates for a data-driven, learning-augmented simulation paradigm in addressing the myriad complexities inherent in container port truck dispatch dilemmas.

Chapter 6

Neural Networks Assisted Methods in Dispatching Strategy Refinement and Evaluation Acceleration

In the evolving digital era, the prowess of neural networks and deep learning has undeniably reshaped the landscape of computational methodologies. Their unparalleled capabilities in learning intricacies and addressing multifaceted problems have heralded a new dawn in myriad sectors. Yet, their inherent "black-box" nature, an obscurity that masks the internal mechanics of their decision-making processes, has often posed challenges in deriving intuitive interpretations. Such opacity has inevitably raised eyebrows, limiting their ubiquitous implementation, especially in applications demanding explicit rationale behind decisions.

One such critical domain is container port truck dispatching—a realm where the imperatives of timeliness, decipherability, and efficacy converge.

Given the magnitude of these requirements, it's perhaps not surprising that neural networks haven't yet fully penetrated this space, despite their evident computational supremacy.

Notably, GP revered for its innate transparency and robust generalizability, has carved a niche for itself, becoming a preferred choice in such demanding scenarios. Moreover, when juxtaposed with RL—a paradigm resembling neural networks—it is evident that the amalgamation with GP acts as a catalyst, enhancing RL's efficiency.

However, it's imperative to dispel the notion that the capabilities of neural networks are orthogonal to the needs of container port truck dispatching. Far from it. This chapter unravels how neural networks can be harmoniously integrated into this domain. By elucidating their application in two pivotal facets of the container port truck dispatching conundrum, we aim to illuminate the path for marrying computational rigor with practical exigencies, fostering a symbiotic coexistence between advanced neural architectures and real-world logistical challenges.

The evolving landscape of computational intelligence is underpinned by integrating diverse methodologies, each with its unique vantage point, converging to address the multifaceted challenges of intricate problems. As we navigate the complex domain of container port truck dispatching, the utility of neural networks emerges as a potent force, particularly when juxtaposed against the established prowess of GP.

In the realm of GP, its strength is undeniably anchored in its adeptness at traversing vast solution spaces, illuminating potential solutions that might remain elusive to more conventional techniques. However, it is not without its constraints. The challenges encountered with GP are its relatively tepid convergence rate in localized optimization scenarios and the occasional in-stability observed during post-evolutionary phases. Such phases can lead to an undesirable genetic uniformity in the population, undermining the rich diversity essential for robust evolutionary outcomes. This scenario unveils a pivotal juncture where the capabilities of neural networks can be seamlessly integrated. By leveraging their granularity in optimization, neural networks can intricately refine the promising solutions identified by GP. Moreover, as these neural architectures modify individual entities, they pave the way for the genesis of novel genotypes, ensuring a diverse and vibrant genetic pool, and thus bolstering the overall quality of evolutionary results.

Transitioning to the facet of creating and evaluating truck dispatching strategies, the unique intricacies inherent to real-world operational scenarios necessitate reliance on simulators. An often underappreciated yet critical determinant in this equation is the simulation speed, which directly influences the efficacy of the truck dispatching strategies. While traditional simulation methodologies offer robustness, they are frequently entangled in many computational complexities, leading to extended evaluation periods. This is where the intricate architectures of neural networks come to the fore. With their innate ability to emulate diverse operational scenarios, they can effectively assess the performance of myriad dispatching strategies. This capability offers a dual advantage: not only do neural networks provide insights that closely mirror real-world outcomes, but they also achieve this in a fraction of the time required by conventional simulators.

Finally, it is imperative to underscore that the discourse surrounding the potential interplay between neural networks and truck dispatching is still nascent. The vast and dynamic terrain of this domain presents a plethora of unexplored avenues and challenges. The synergistic melding of neural networks with sophisticated dispatching paradigms heralds a new era, ripe for exploration and innovation, beckoning researchers to delve deeper, unearthing transformative strategies that promise to reshape the contours of logistics and computational intelligence.

6.1 Neural Networks in Truck Dispatching Strategy Refinement

6.1.1 Recurrent Neural Network

RNNs have been increasingly recognized for their ability to process sequence data, making them particularly suitable for producing heuristics in dynamic dispatching (Klug et al., 2019).

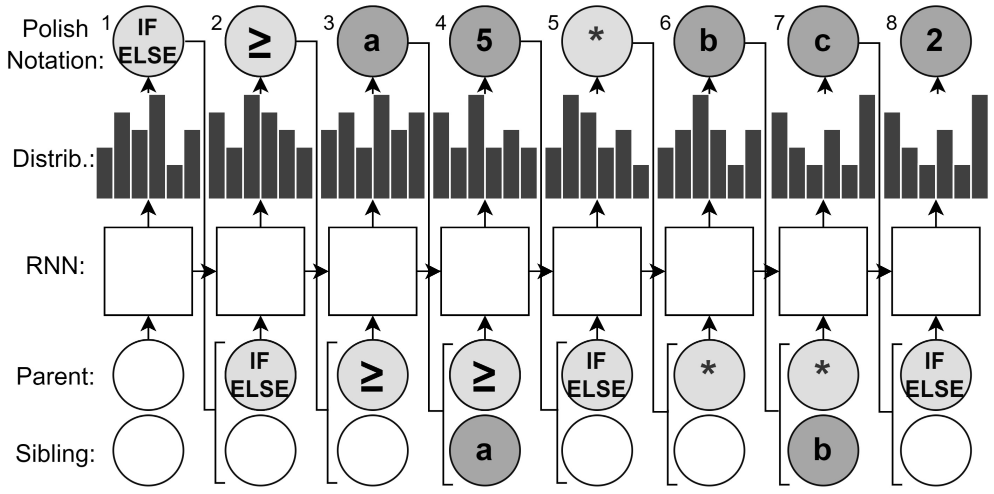


Figure 6.1: RNN Workflow

Like GP, RNN also produces heuristics for truck dispatch, but it employs a different representation. The RNN sequentially forms a heuristic, represented as a list using Polish notation. For instance, a LGP tree can corresponds to $[if\_else, \geq, a, 5, *, b, c, 2]$. As shown in Fig. 6.1, to generate this list, the RNN calculates probabilities for choosing different operators and parameters, considering both parent and sibling nodes. A token is then randomly selected based on these probabilities, leading to the step-wise construction of the Polish notation, or heuristic. Logical operators from the LGP framework have been integrated into the RNN model to improve its performance. Additionally, the RNN model includes an "End Token," which differs from other tokens. Once this token is encountered, the RNN ceases generation. This specific token enables the RNN model to produce subtrees, allowing for variable tree sizes instead of generating exclusively full trees.

Commonly, an RNN is selected such that the parameters θ render the likelihood of an expression tractable, thereby enabling the back-propagation of a differentiable loss function. For the ith token, denoted as τ_i, its likelihood is conditionally independent given the preceding tokens $\tau_1, ..., \tau_{(i-1)}$. Therefore, $p(\tau_i|\tau_{j\neq i}, \theta) = p(\tau_i|\tau_{j<i}, \theta)$.

This study implements the RNN structure as informed by prior research (Mundhenk et al., 2021), specifically adopting an auto-regressive RNN constituted of a single-layer Long Short-Term Memory (LSTM) with 32 hidden nodes. Concurrently, the traditional Vanilla Policy Gradient (VPG) method has been utilized to train the RNN in this research.

VPG employs the well-established reinforce rule, with training conducted over the batch T. This results in the following loss function: $L(\theta) = \frac{1}{|\tau|}\sum_{\tau\in T}(R(\tau) - b)\nabla_\theta log(p(\tau|\theta))$, where b is a baseline term or control variate, such as an exponentially-weighted moving average of fitness.

The fitness function used to train the RNN is defined as $reward = (maxE - minS) - \Delta cov(O_r, O_m)$, where $maxE - minS$ represents port operation efficiency, Δ is a weight factor set to one in this study, cov is a covariance function, O_r is the task ranking produced by RNN, and O_m is the ranking given by the aforementioned manual heuristics. This function aims to increase the dispatching efficiency of the RNN while also guiding the RNN to mimic the learning of the manual heuristic, thus enhancing both performance and convergence efficiency.

Our tests indicate that when used alone, the RNN does not provide optimal solutions to the port truck dispatching problem. This is due to its difficulty in establishing an effective starting position for search, which restricts the full potential of its excellent local search capabilities. To address this issue, we propose the Neural Network Assisted Genetic Programming (Neural Network Assisted Genetic Programming (NN-GP)) model which promotes cooperative interaction between GP and RNN to enhance overall performance.

6.1.2 Neural Network Assisted Genetic Programming

While both GP and RNN possess distinct advantages, their standalone application also presents certain limitations. Consequently, we propose the NN-GP approach in this section. This integrated method aims to amalgamate the strengths of both GP and RNN while simultaneously mitigating their individual shortcomings.

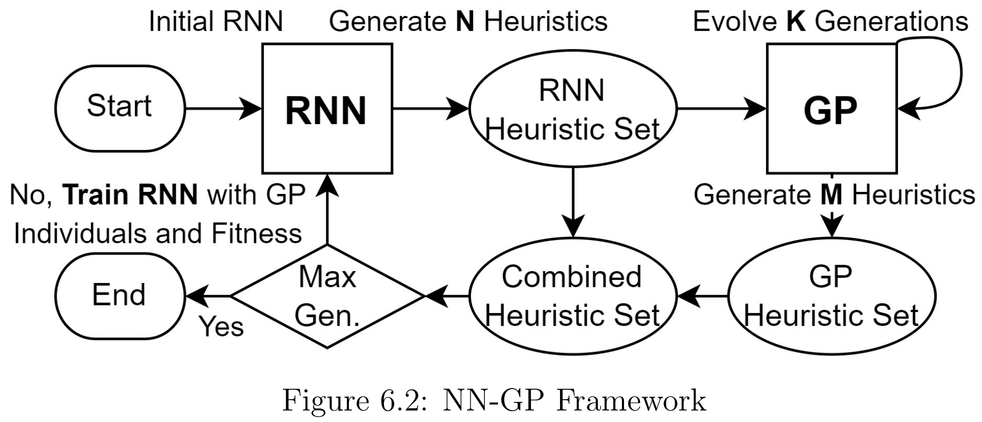


Figure 6.2: NN-GP Framework

The framework of the NN-GP model, as illustrated in Fig. 6.2, proceeds as follows: Firstly, the RNN—initialed with random neural networks, both logical and arithmetic operators, and feature parameters—follows the procedure depicted in Fig. 6.1 to construct N heuristics, which subsequently act as the initial population for the forthcoming GP processes. According to the findings in Mundhenk et al. (2021), optimal performance is typically achieved when M is twice the size of N. Furthermore, the remainder of the GP individuals are initialized using the traditional ramped half-and-half method in this section. After K generations of training predicated on this initial population, the GP yields M heuristics. The N heuristics initially generated by the RNN are then amalgamated with the M heuristics derived from the GP. If the predetermined maximum number of training generations is attained, the training process halts. Otherwise, the fitness for each heuristic in the merged set is calculated. Subsequently, the RNN is trained using these computed fitness values and the corresponding heuristics. This training allows the RNN to continuously adjust the probability distribution of its output across diverse inputs, thereby fostering the generation of improved heuristics. Following this step, the RNN generates N new heuristics for the next cycle.

The hybrid NN-GP model presents several key advantages that bolster both the efficacy of the training process and the quality of the solutions:

- **Population Diversity:** The integration of RNN enhances the diversity of the GP populations, promoting a more efficient evolutionary process.
- **Local Search Capabilities:** RNN augments GP with strong local search capabilities. By effectively exploring the solution landscape around promising areas identified by GP, RNN can uncover solutions

that GP's broader, population-based search might miss.

- **Complementarity:** GP and RNN complement each other, leading to an improvement in the overall quality of the solution. GP's aptitude for broad, population-based search is supplemented by RNN's expertise in refined local search, resulting in a comprehensive exploration and exploitation of the solution space.

In summary, the NN-GP model harnesses the strengths of both GP and RNN, culminating in a robust, efficient, and versatile tool for addressing the dynamic truck dispatching problem.

6.1.3 Experiment and Discussion

This section is devoted to assessing the performance of AGP, LGP, RNN, and NN-GP in addressing a complex container port dispatching problem characterized by uncertain environmental parameters. These methods are juxtaposed against the standalone GP and RNN method, with NN-GP exemplifying the benefits derived from the collaboration of GP and RNN. Furthermore, considering the stable performance and successful implementation of the manual heuristic, we opt to use it as a benchmark for comparison.

The environmental parameters, or GP/RNN terminals, used in this study adhere to those defined in our previous research (Chen et al., 2022). A total of 14 features—including truck travel time, the current number of QC trucks, the number of waiting trucks, the number of remaining tasks, and other pertinent characteristics—are utilized to depict the current operational state of the container port. The GP features a population size of 1024, with crossover, mutation, and reproduction rates set at 60%, 30%, and 10%, respectively. The fitness function utilized within the GP is denoted by equation (1.4). All algorithms are subjected to 300 training generations, with the RNN's learning rate fixed at 0.001. In alignment with the NN-GP framework delineated in Fig.6.2 and following the guidance of Mundhenk et al. (2021), K is established at 20, indicating the involvement of RNN every 20 GP training iterations. M equals the GP population size of 1024, and the N value is set at half of M, which is 512.

The datasets used in this experiment are the same as our previous research (Chen et al., 2022), and were procured from historical operational data at the Ningbo Port. The port scenario encompasses a single ship berth with six QCs. The number of trucks is in accordance with the actual number of working trucks during the data extraction period, varying between 24 and 48. Variables such as truck travel time and container load and unload times are computed based on real operation time distributions, characterizing them as uncertain variables.

Historical task data from different time periods were collected into 10 distinct sets, mirroring the diverse operational scenarios encountered at various times. Of these sets, half were employed for training purposes, while the remaining half were reserved for testing. Each set for training or testing contains 10 instances from the same time period, encompassing 200 tasks comprising a mixture of loading and unloading operations. For each training instance, 100 independent runs were conducted with distinct random seeds. The average training and testing results over these 300 runs for each set are presented in Table 6.1.

Owing to the disparate benchmark performances on real data, we introduce the improvement over the manual heuristic (Imp.) as a baseline, with all subsequent comparisons illustrating the improvement over this baseline.

Table 6.1: AGP, LGP, RNN and NN-GP Experiment Results (TEU/h)

Set	**No.**	**Manual**	**AGP**	**LGP**	**RNN**	**NN-GP**[1]	**NN-GP**[2]
Train	1	106.87	114.37	122.70	106.19	116.37	126.15
	2	118.91	125.18	131.36	123.64	129.25	134.78
	3	114.27	122.81	125.98	117.65	124.59	128.21
	4	121.48	131.68	135.59	122.15	132.96	143.49
	5	116.88	126.47	125.92	119.93	134.49	138.37
	Avg.	115.68	124.10	128.31	117.91	127.53	134.20
	Imp.	**0.00%**	**7.28%**	**10.92%**	**1.93%**	**10.24%**	**16.01%**
Test	1	105.35	104.06	113.54	104.56	110.88	115.07
	2	126.85	136.19	136.69	123.89	128.10	135.64
	3	115.72	120.20	125.98	117.44	122.77	129.49
	4	106.01	112.89	117.34	108.03	112.98	114.74
	5	121.37	128.93	126.03	119.05	127.27	138.21
	Avg.	115.06	120.46	123.92	114.59	120.40	126.63
	Imp.	**0.00%**	**4.69%**	**7.70%**	**-0.41%**	**4.64%**	**10.06%**

[1] This refers to the pared-down version of NN-GP, which solely incorporates the Arithmetic operators.

[2] This refers to the comprehensive version of NN-GP, enriched with the inclusion of the Logic operators.

Experimental results indicate that LGP incorporating logical operators outperforms AGP on training and test sets. Through a t-test on the experiment ($\alpha = 0.001, p = 0.00$), the full version of NN-GP further improves performance compared to either GP or RNN alone, surpassing LGP by 5.09% and 14.08% on the training and test sets, respectively. Conversely, RNN alone exhibits inferior performance, with only a 1.93% improvement on the test set and even worse results than the artificial heuristic on the test set. These findings suggest that both GP and RNN individually underperform compared to NN-GP, indicating that NN-GP, by leveraging the strengths of both RNN and GP, achieves enhanced performance and more effectively addresses the dynamic port assignment problem.

To investigate the reasons behind NN-GP's superior performance, we removed the logical operators from the full NN-GP and observed a substantial drop in performance. Although its performance on the training set was similar to that of LGP, its test set performance deteriorated significantly, resembling that of AGP. This highlights the contribution of incorporating logical operators into NN-GP. In addition, we evaluated the impact of fitness by employing a fitness function, $fitness = (maxE - minS)$, which solely considers the port work efficiency. The findings reveal that this modification led to an approximate decrease of 17% in the average performance of both RNN and NN-GP. This observation underscores the efficacy of our newly designed fitness function in accelerating the convergence speed of RNN and NN-GP, whilst concurrently enhancing their performance.

In summary, the NN-GP model effectively integrates the strengths of RNN and GP, demonstrating superior performance in both training and test scenarios. It adeptly handles the intricacies of dynamic truck dispatching in multi-scenario ports, despite inherent uncertainties. Given its successful application, the model shows considerable potential for extension to other dynamic transportation optimization problems, thus indicating a wide scope for future applications.

6.1.4 Conclusion

In conclusion, this research introduces the novel NN-GP model, a combination of GP and RNN, to tackle the complex dynamic truck dispatching problem. The combination of GP's broad exploration of the solution space and RNN's superior local search capabilities ameliorates the limitations of the standalone methodologies. Additional enhancements, such as the integration of logic operators and the design of a new fitness function, further expedite convergence speed and elevate the overall performance of the NN-GP model.

Experimental results robustly attest to the effectiveness and efficiency of the NN-GP approach. Compared to individual applications of GP, RNN, and traditional dispatching methods, the NN-GP model exhibits superior performance and stability, particularly in complex, multi-scenario container port dispatching problems characterized by uncertainty. The model's reusability and quick adaptability to new operating scenarios underscore its potential for real-world implementation in dynamic environments.

While this study constitutes major progress in applying hybrid methodologies to logistical problems, future research could delve deeper into potential enhancements to the NN-GP model. This could include the integration of additional machine learning techniques or the creation of a more refined fitness function, further expanding the model's capabilities in addressing dynamic and uncertain operational environments.

6.2 Neural Networks in Truck Dispatching Simulation Acceleration

6.2.1 Transformer Surrogate Model

The limitations inherent in conventional GP implementations, specifically regarding the restricted number of evolutionary generations in unit time and frequent converging failures, necessitate a more efficient approach to evaluating individual fitness. To address this challenge, we introduce a transformer surrogate model as a viable alternative to the traditional simulation based fitness evaluator.

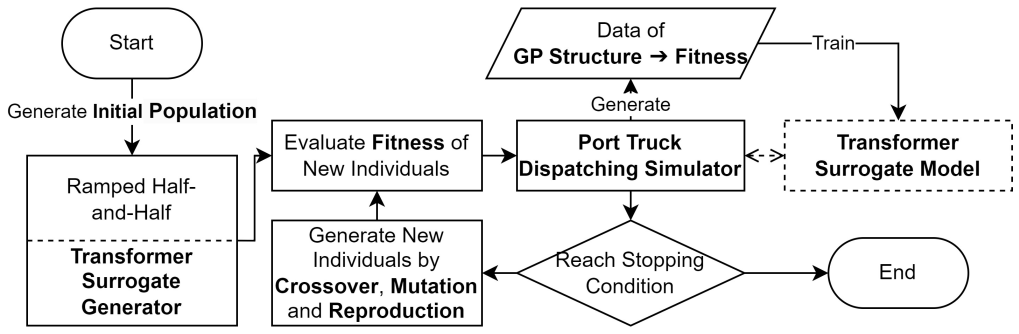

Start
Generate Initial Population
Ramped Half-and-Half
Transformer Surrogate Generator
Evaluate Fitness of New Individuals
Generate New Individuals by Crossover, Mutation and Reproduction
Data of GP Structure ➔ Fitness
Train
Generate
Port Truck Dispatching Simulator
Transformer Surrogate Model
Reach Stopping Condition
End

Figure 6.3: Transformer Surrogate GP Framework

A transformer model, initially proposed for natural language processing tasks (Vaswani et al., 2017), has demonstrated its versatility and efficacy across various domains. In this study, the transformer model is a surrogate for the conventional simulation, providing rapid and accurate evaluations of GP individual candidates. As shown in Fig. 6.3, in the process of GP evolving, all newly generated individuals have to go through the port truck dispatching simulator to compute their fitness. Even though this part of the work can be calculated in parallel using multi-threaded computation, it still consumes many computational resources. Therefore, we propose to use a transformer surrogate model instead of the port truck dispatching simulator to accelerate the GP training. The transformer model in the paper follows the default model of the Transformers (Wolf et al., 2020a) except for changing the expected and actual output to one double float. According to our experimental results, using the transformer surrogate model can reduce the fitness computation time by more than 65% and is not affected by the increased number of tasks.

For the training of the transformer surrogate model, we propose to use the fitness data corresponding to the GP individual structure generated during the GP evolution process to train the transformer directly so that it does not need to consume time to produce data to prepare the transformer. The training process of the transformer can be carried out simultaneously with the GP evolution process, so in this paper, training the transformer surrogate model does not increase the training time of the algorithm. The transformer training process can be carried out simultaneously with the GP evolution process, so in this paper, training the transformer surrogate model does not increase the training time of the algorithm.

In addition, for the trained transformer surrogate model to perform well on datasets with different operating environments and to eliminate the interference of operating environments, the transformer surrogate model will not directly output the same TEU/h data as the simulator but will instead generate a number between 0 and 1, which represents the corresponding size of the GP individual's fitness. Once trained, the model estimates the fitness of new GP individuals with a substantially reduced computational cost.

6.2.2 Transformer-Surrogate Genetic Programming

After constructing the transformer surrogate model, we integrate it with GP to formulate the Transformer-Surrogate Genetic Programming (Transformer-Surrogate Genetic Programming (TSGP)) algorithm. As delineated in Algorithm 7, the evolutionary process of TSGP is carried out in multiple stages. Initially, up to generation x, fitness values for GP individuals are computed using a traditional simulator, while simultaneously training the transformer surrogate model with the newly generated data. Subsequently, for y generations, the fitness computation shifts to the transformer surrogate model. After this, the algorithm reverts to the simulator for z generations, updating the surrogate model in tandem. This loop continues unless the termination criteria are met.

Algorithm 7 Transformer-Surrogate Genetic Programming Evolving

```
1: Initialize: GP Population, Transformer Surrogate Model
2: $t \leftarrow 0$                                  ▷ Initialize training generation counter
3: while Training time or number of generations not reached do
4:     if $t < x$ then                                        ▷ First $x$ generations
5:         Compute Fitness of GP individuals using Simulator
6:         Update Transformer Surrogate Model with Individual - Fitness
   Data
7:     else if $t \mod (x + y + z) < x + y$ then          ▷ Next $y$ generations
8:         Compute Fitness of GP individuals using Transformer Surrogate
   Model
9:     else                                                   ▷ Next $z$ generations
10:        Compute Fitness of GP individuals using Simulator
11:        Update Transformer Surrogate Model with Individual - Fitness
   Data
12:    end if
13:    Perform GP operations (Selection, Crossover, Mutation)
14:    $t \leftarrow t + 1$                                 ▷ Increment generation counter
15: end while
```

The hyperparameters x, y, and z can be adjusted based on the complexity of the problem and the optimization goals. Our exploratory experiments indicate that setting y to be ten times z results in improved performance.

Additionally, x should ideally be greater than 100 to ensure sufficient data for training the surrogate model. Consequently, this study empirically establishes these parameters at 100, 1000, and 10, respectively. Nonetheless, determining the optimal hyperparameter settings for NN-GP remains a critical issue, warranting in-depth investigation in future research.

Building upon TSGP, we extend the algorithm to TSGP*, an enhanced version that leverages the trained transformer surrogate model for generating a more effective initial population. Unlike the traditional ramped half-and-half initialization method in standard GP, TSGP* employs a transformer surrogate generator. As elaborated in Algorithm 8, this generator also uses the ramped half-and-half technique to produce initial GP individuals. However, following their generation, the trained transformer surrogate model is invoked to estimate their fitness values. Only those individuals surpassing a fitness threshold f are retained. The value of f is a tunable hyperparameter set to 0.5 in this study.

Algorithm 8 Transformer Surrogate Generator

```
 1: Initialize: Empty GP Population, Transformer Surrogate Model
 2: $m_{\text{count}} \leftarrow 0$                        ▷ Initialize individual counter
 3: $f \leftarrow$ Fitness threshold
 4: while $m_{\text{count}} < m$ do                  ▷ Until $m$ individuals are generated
 5:     individual ← Generate GP tree using ramped half-and-half
 6:     fitness ← Evaluate fitness using Transformer Surrogate Model
 7:     if fitness $> f$ then
 8:         Add individual to GP Population
 9:         $m_{\text{count}} \leftarrow m_{\text{count}} + 1$
10:     end if
11: end while
```

In the subsequent section, we shall empirically assess the efficacy of both TSGP and TSGP* algorithms by applying them to a real-world case study involving dynamic truck dispatching in container ports. This performance evaluation will juxtapose multiple state-of-the-art methodologies, facilitating a comprehensive comparative analysis.

6.2.3 Experiment and Discussion

This section is devoted to a comprehensive evaluation of a diverse array of algorithms, specifically Data-Driven Genetic Programming (GP) (Chen et al., 2020), Deep Reinforcement Learning Hyper-Heuristic (DRL-HH) (Zhang et al., 2022), Neural Network Assisted Genetic Programming (NN-GP) (Chen et al., 2023), TSGP, and TSGP*. The primary focus rests on elucidating the performance merits of TSGP and TSGP*, emphasizing the collaborative efficacy of amalgamating GP and transformer surrogate models. Given its robust and empirically validated performance, the manual heuristic (Chen et al., 2016) is the chosen benchmark for this comparative study.

The feature set used in this study, often termed GP terminals, closely follows the specifications outlined in our prior research (Chen et al., 2022). It encompasses 14 distinctive features, such as truck travel time, the current quantity of QC trucks, the number of trucks in waiting, and the remaining task count, among other salient variables, to accurately portray the operational state of the container port at any given moment. The GP algorithm is configured with a population size of 1024 and employs crossover, mutation, and reproduction rates of 60%, 30%, and 10%, respectively. All the algorithms in this study were subjected to a 10-hour training regimen.

The datasets leveraged in this experiment are consistent with those used in our previous work (Chen et al., 2022), originating from historical operations at Ningbo Port. The experimental setup mimics a real-world single ship berth featuring six QCs, with the number of trucks being variable and reflecting the actual operational conditions, ranging between 24 and 48. Uncertain variables like truck travel time and container loading and unloading times are modeled based on empirical distributions derived from genuine operational data.

For this study, historical task records were categorized into ten unique sets, each representing a distinct operational scenario encountered at different temporal intervals. Five of these sets were used for training, while the remainder were for testing. Each training or testing set consists of ten instances sourced from a similar time frame, incorporating 200 tasks that include a blend of loading and unloading operations. Each training instance was executed 100 times, employing different random seeds for each run. A comprehensive summary of the average training and testing results is provided in Table 6.2.

Table 6.2: GP, DRL-HH, NN-GP, TSGP and TSGP* Experiment Results (TEU/h)

Set	**No.**	**Manual**	**GP**	**DRL-HH**	**NN-GP**	**TSGP**	**TSGP***
Train	1	106.87	118.76	117.17	117.84	122.05	130.84
	2	126.85	133.84	139.85	145.21	147.99	149.01
	3	114.27	120.76	127.00	126.14	132.30	134.36
	4	106.31	113.78	112.91	117.14	123.55	122.31
	5	116.88	129.17	125.75	132.60	133.90	136.91
	Avg.	114.236	123.26	124.53	127.79	131.96	134.68
	Imp.	**0.00%**	**7.90%**	**9.02%**	**11.86%**	**15.51%**	**17.90%**
Test	1	105.87	113.05	115.01	111.19	118.70	119.46
	2	118.91	125.93	126.59	134.01	135.50	133.47
	3	115.72	122.58	124.48	126.06	133.81	133.78
	4	121.48	130.10	132.59	134.92	135.93	139.37
	5	117.25	120.63	125.28	127.15	135.72	132.23
	Avg.	115.846	122.46	124.79	126.67	131.93	131.66
	Imp.	**0.00%**	**5.71%**	**7.73%**	**9.34%**	**13.89%**	**13.65%**

In light of varying benchmark performances observed in real-world data, we introduce the Improvement over the Manual Heuristic (Imp.) as a reference baseline. All subsequent performance metrics are framed as improvements over this established baseline. A t-test conducted on the experimental outcomes ($\alpha = 0.001, p = 0.00$) substantiates that the fully realized TSGP and TSGP* variant exhibits notable performance enhancements over GP, DRL-HH, and NN-GP counterparts—registering an improvement of approximately 7%, 5%, and 3% on the training and test sets, respectively. Notably, the TSGP model demonstrates negligible performance degradation when transitioning from training to test sets. This underscores TSGP's robustness and effective generalization capabilities, rendering it highly applicable to previously unseen datasets.

To explore TSGP's capabilities further, we assessed TSGP*, a variant that integrates a transformer-based surrogate GP generator. TSGP* is designed to outperform its predecessor by utilizing a surrogate model to select the initial GP population, theoretically enhancing its quality and efficacy. However, while TSGP* demonstrates improved performance on the training set, its effectiveness diminishes on the test set, suggesting an overfitting issue. This problem arises because the surrogate model, trained on the training data, tends to favor individuals that excel on the training data, thereby reducing the genetic diversity of the GP population and adversely affecting test performance.

This outcome underscores the need for further research to address TSGP*'s overfitting challenge. One potential solution could involve introducing a more diverse range of GP individuals during TSGP*'s evolution process to mitigate the diversity shortfall. Future research could delve deeper into the utilization of surrogate models for initializing GP populations. This area holds significant promise for enhancing the efficiency and effectiveness of genetic programming by potentially streamlining the selection of high-quality initial individuals after handling the overfitting problem.

In conclusion, the TSGP framework adeptly amalgamates the strengths of both the transformer and GP paradigms, showcasing superior performance in varied training and testing conditions. The model addresses the high simulation cost inherent in dynamic truck dispatching across multi-scenario ports, even amidst uncertainties. Given its successful implementation, TSGP holds substantial promise for broader applications in dynamic transportation optimization problems, opening up a wide avenue for future research endeavors.

6.2.4 Conclusion

This study introduces the TSGP approach, a groundbreaking methodology that addresses the computational inefficiencies in traditional GP applied to truck dispatching in container ports. By employing a transformer model as a surrogate evaluator during the fitness calculation stage, TSGP not only drastically reduces the algorithmic training time but also enhances the performance of GP. The transformer model further serves a dual role by generating optimized initial populations for GPs, demonstrating a synergistic integration of the computational strengths of transformer models with the heuristic search capabilities of GPs.

Our empirical results confirm the efficacy of TSGP, especially its ability to significantly accelerate the training process while maintaining or improving operational efficiency. The transformer model effectively learns to approximate the fitness landscape of the GP, thereby streamlining the evolutionary process. However, while TSGP and TSGP* showed robust performance in both training and testing scenarios, some challenges related to initial population diversity in our extended model, TSGP*, indicate directions for future research.

In summary, the TSGP approach marks a substantial advance in container port optimization, particularly in truck dispatching systems. By marrying the capabilities of transformer models and GPs, we open new avenues for applying machine learning techniques in operational logistics and beyond. The demonstrated success of TSGP and TSGP* sets the stage for their broader application across various domains where computational efficiency and effective heuristic search are critical.

6.3 Summary

In this chapter, we have embarked on an enlightening journey through the intersection of intricate logistical problems and state-of-the-art computational methodologies. Through this exploration, two promising models emerge, namely the NN-GP model and the TSGP approach, both tailored to address the dynamic truck dispatching problem in container ports.

The NN-GP model represents a harmonious fusion of GP and RNN. By capitalizing on the extensive solution space exploration capabilities of GP and the nuanced local search prowess of RNN, the NN-GP model successfully overcomes the limitations inherent to each methodology when applied in isolation. Augmenting this combination is the integration of logic operators and the innovative design of a new fitness function, both of which significantly bolster the convergence speed and the model's overall efficacy. On the other hand, the TSGP approach, while bearing its roots in GP, introduces a transformative change by harnessing the computational strengths of transformer models. These models not only accelerate the GP training process but also function as invaluable tools in generating optimized initial populations for GPs, embodying a seamless amalgamation of heuristic search with cutting-edge machine-learning capabilities.

Empirical evaluations conducted during this study provide compelling evidence supporting the robustness, efficiency, and adaptability of both models. In juxtaposition with traditional dispatching methods, standalone GP, and RNN applications, the NN-GP model and the TSGP approach consistently demonstrate superior performance, stability, and resilience, especially in intricate, multi-scenario container port dispatching environments riddled with uncertainties. The models' ability to rapidly acclimatize to evolving operating scenarios further solidifies their prospects for deployment in real-world, dynamic logistical settings.

However, the journey of exploration is seldom without challenges. While the TSGP and its extended version, TSGP*, have shown commendable results in diverse scenarios, issues related to initial population diversity in TSGP* furnish cues for subsequent research endeavors.

In synthesizing the insights from both models, it becomes palpable that the convergence of advanced machine learning techniques with heuristic algorithms heralds a promising frontier in logistical optimization. The successes delineated by the NN-GP and TSGP models underscore the vast potential of such hybrid methodologies in reshaping the future of operational logistics and computational intelligence.

Regarding the interpretability of the learned models, RNN and Transformer models are inherently challenging to interpret due to their reliance on complex deep neural network architectures. Despite their effectiveness, both models lack straightforward interpretability. Conversely, the TSGP and NN-GP methods introduced in this chapter are more comprehensible, thanks to their integration with the inherently interpretable GP method. This fusion of machine learning models with GP, an interpretable meta-heuristic approach, offers innovative pathways for developing interpretable

AI solutions.

As we conclude this chapter, we remain optimistic about the potential extensions, refinements, and broader applications of these models, confident that they will continue to pave the way for breakthroughs in domains where efficiency, adaptability, and innovation are paramount.

Chapter 7

Conclusion and Future Work

As we draw this thesis close, we must revisit our journey, delving into the myriad applications of machine learning within container port truck dispatching. From its outset, this research explored how machine learning, with a particular focus on Genetic Programming, can be harnessed to reimagine and optimize processes within the intricate tapestry of port operations. Our narrative has spanned from enhancing operational efficiency at ports and augmenting the accuracy of simulations to the strategic deployment of neural networks for truck dispatching, demonstrating the extensive applicability of machine learning across diverse port functionalities.

Given the unique characteristics and demands of real-world port environments, this thesis emphasized GP methodologies inspired by decision trees to address truck dispatching dilemmas. While exhibiting potential for widespread adoption, such an approach undeniably encounters performance constraints. However, it's imperative to recognize the transformative combination birthed when GP is fused with various advanced machine learning techniques. By integrating methods such as Ensemble Learning, Reinforcement Learning, Recurrent Neural Networks, and Transformers, the revamped GP retains its merits of interpretability and computational efficiency and experiences a significant elevation in its performance metrics for truck dispatching problems.

This concluding chapter aims to encapsulate the entirety of the thesis, offering a synthesized overview of the content and spotlighting the seminal contributions made. It is designed to reflect coherently on the strides taken, challenges encountered, and knowledge advanced. Beyond merely cataloging the achievements, this chapter will underscore the limitations observed and pave the way by outlining potential directions for future research.

When aligning our retrospective gaze with prospective aspirations, we intend to validate this research's endeavors and position it as a foundational stepping stone, inspiring further academic and industry-driven inquiries into the expansive domain of machine learning-infused container port operations.

In this landscape of rapid technological advancements, the potent amalgamation of machine learning methods and logistical operations emerges as both an opportunity and a necessity. The port industry, characterized by its dynamic environment and the ever-present imperative for efficiency, provides a fertile ground for such interdisciplinary explorations. Having charted the course of this intricate fusion over the preceding chapters, we find ourselves poised to distill the essence of our findings and their implications for academia and industry.

7.1 Summary of The Study

As the world of port logistics pivots toward ever-increasing efficiency and throughput, this thesis embodies that transformative aspiration. This extensive investigation was borne from recognizing a quintessential challenge: synchronizing containerized freight's burgeoning demands with global ports' intricate operational capabilities. The research trajectory charted here sought to merge the frontiers of multiple disciplines, thus providing an interdisciplinary vista into the realm of container port operations.

At the heart of this research lies the crucial operation of container truck dispatching. Recognizing its significance, we embarked on a journey to understand, enhance, and optimize the strategies that govern this essential function. The container truck, as this thesis illustrates, isn't merely a vehicle—it's a conduit that bridges the vast infrastructural expanse of a port with the relentless rhythm of global trade. Therefore, the efficiency with which it operates has ripple effects, impacting everything from economic viability to the entire port's work efficiency.

The cornerstone of our approach involved the innovative fusion of traditional logistical know-how with cutting-edge machine learning methodologies. We ventured beyond the conventional, transitioning from manual crafted heuristic methods that have traditionally dominated the domain to advanced algorithmic paradigms, notably Genetic Programming and Reinforcement Learning. This evolution represents not just a technological advancement but a paradigm shift. By harnessing the power of these machine learning techniques, the research unveiled the potential for automatically curating dispatching strategies that are both efficient and adaptable.

One of the salient features of our exploration was its adaptability. Traditional heuristic methods, despite their provenance, are inherently rigid. They often lean heavily on the expertise of seasoned professionals, whose invaluable insights are not always scalable or adaptable to every nuanced change in the operational context. Our foray into machine learning transcended this limitation. The resultant methodologies exhibited a versatility that was hitherto uncharted. They demonstrated an ability to leverage vast and intricate data patterns, synthesizing them into suitable dispatching strategies for various scenarios.

Additionally, our study addressed the complex task of simulating port traffic environments, particularly in scenarios where GPS data may be imprecise. By merging the self-learning attributes of GP and RL, the fidelity of truck dispatching simulation in sparse data scenarios was notably enhanced, especially at intersections within the harbor. Incorporating transformer-based techniques in our research marked a significant advancement in simulation accuracy and speed. This not only paved the way for devising strategies that are efficient but also remarkably precise.

Beyond the technical and operational dimensions, the research also illuminated the broader impacts of optimized dispatching. The study enriched the discourse on port logistics by mapping the intricate correlations between efficient dispatching, judicious resource utilization, sustainability, and overall operational throughput. This wasn't just about moving containers more swiftly; it was about understanding how every operational enhancement resonated across the economic, environmental, and organizational fabric of port operations.

The thesis represented a synthesis of insights from various fields: computer science, operations research, industrial engineering, and environmental studies. It offered a kaleidoscopic view into the world of port operations, capturing its multifaceted challenges and presenting algorithmic solutions that were efficient, sustainable, and environmentally conscious.

In this thesis, we emphasized solutions applicable to real-world scenarios and introduced the use of GP to generate interpretable heuristics resembling manual heuristic decision trees to address the problem. This approach met the requirements for interpretability in practical port operation settings and enabled swift dispatching during equipment downtime. Addressing the generalization limitations of heuristic methods, such as average performance and weak adaptability to different environments, we proposed various structures combining GP with GP, GP with RL, and GP with (NN. While retaining the advantages of GP-based heuristics, these structures substantially enhanced their performance, presenting new directions for optimizing container port truck dispatching and other multifaceted scenarios.

On another front, we introduced learning-based methods for intersection simulation to address the accompanying port simulation challenges in container port truck dispatching. This significantly improved the accuracy of port simulations. Furthermore, we proposed using the Transformer architecture to construct surrogate models, accelerating port simulations and achieving commendable results.

The research journey, organized meticulously across multiple chapters, started with an extensive literature review, encapsulating the existing body of knowledge on container port logistics, dispatching, and the burgeoning role of machine learning. Subsequent chapters delved deep into specific methodologies, from Genetic Programming to Reinforcement Learning, exploring their implications, methodologies, and contributions. The narrative culminated in a comprehensive conclusion, synthesizing the multifarious insights gleaned and paving the way for future explorations.

In summation, this study stands as a testament to the transformative potential of interdisciplinary research. It pushed the boundaries of what's possible in container port logistics and forged new pathways, intertwining machine learning, environmental sustainability, and operations research. The thesis has contributed profoundly to academic discourse and real-world operational paradigms in port logistics through its nuanced exploration of container truck dispatching and its broader implications.

7.2 Contributions to The Field

The port logistics and operations sphere has been an evolving area of study, with constant strides made toward improving efficiency, sustainability, and overall operational excellence. This research has aimed to add to this collective body of knowledge and carve out new avenues and perspectives that could redefine paradigms in container port operations. The contributions to the field from this research can be delineated as follows:

Innovative Integration of Machine Learning: While introducing technology into port operations isn't novel, this thesis heralds the seamless and pioneering integration of advanced machine learning methodologies into the domain of container truck dispatching. The research has established a new standard for automated, adaptive, and efficient dispatching strategies by leveraging Genetic Programming, Reinforcement Learning, and other algorithmic paradigms.

Reimagining Traditional Methods: By juxtaposing traditional heuristic approaches with machine learning techniques, this study has highlighted the limitations of relying solely on manual strategies and the advantages of algorithmic ones. This comparison is a blueprint for future research and practical implementations, promoting a shift towards more automated, data-driven methods.

Enhanced Simulation Techniques: One of the standout contributions has been the innovative use of learning-based methodologies for simulating port traffic environments. This advancement has addressed the lacunae in previous simulation methods, especially in contexts where precise GPS data might be limited, thereby improving the accuracy of formulating dispatching strategies. Moreover, the transformer-based model presented in this thesis demonstrated how neural networks can enhance simulation speed.

Holistic Operational Insights: This research extends beyond technical achievements to comprehensively understand how operational efficiency and resource utilization are interconnected. The explainability of the GP-generated model offers operators a new avenue for gaining insights, moving beyond traditional methods. This approach illuminates alternative strategies for understanding port operations, emphasizing the value of learning from machine-generated models.

Interdisciplinary Combination: By amalgamating insights from computer science, industrial engineering, operations research, and environmental studies, the thesis has emphasized the importance and potential of interdisciplinary research in port logistics. This cross-disciplinary approach stands as a template for future studies, advocating for a comprehensive perspective when addressing challenges in the field.

Impact on Organizations: The methodologies proposed in this thesis significantly impact port operations in various aspects. Port operations have become more intelligent by integrating machine learning and metaheuristic-based dynamic dispatching approaches. This advancement aids ports in achieving an intelligent transformation, effectively utilizing the data generated during port production activities. It reduces the excessive reliance on experienced operators, decreases the workload on operators, and enhances operational efficiency in port settings.

Explainable Decision Process: Unlike traditional machine learning methods that rely on deep neural networks and carry the stigma of being "black-box" and uninterpretable, the hybrid machine learning approaches using GP proposed in this thesis do not suffer from these drawbacks. Even when combined with inherently uninterpretable models such as RNNs, RL, and transformers, the GP-based methods retain a degree of interpretability. This feature is indispensable in practical applications where decisions must be substantiated and traceable. GP-based approaches enable the identification of the rationale behind each decision, allowing for the tracing of errors when incorrect decisions are made. This capability ensures that similar issues can be avoided, making these methods particularly valuable for operational transparency and accountability.

In conclusion, this research has bequeathed the field of port logistics with groundbreaking methodologies, fresh perspectives, and a new direction. The insights derived have the potential to revolutionize operational paradigms, enhance economic viability, and foreground environmental sustainability, making this study a pivotal addition to the ever-evolving discourse on port operations and logistics.

7.3 Limitations and Future Research Directions

While pushing the boundaries of existing knowledge, every research comes with its own set of limitations and opens doors to potential avenues of future exploration. Reflecting upon the journey undertaken in this thesis, several limitations and prospective directions for future research can be identified.

Limitations: Data Constraints: The machine learning methodologies implemented in this study are contingent upon substantial and representative data availability. While the models demonstrated effectiveness, their accuracy and adaptability are bound to the quality and comprehensiveness of the data fed into them.

Generalizability: While tested in specific operational scenarios, the methodologies may not guarantee uniform performance across all ports globally. Factors like regional regulations, varying levels of technological adoption, and cultural differences in port operations might affect the applicability of findings.

Algorithmic Complexity: Advanced machine learning techniques, particularly Genetic Programming and Neural Networks, can introduce computational complexities. There might be operational scenarios where real-time decision-making is crucial, and these complexities could introduce delays.

Interdisciplinary Challenges: While the interdisciplinary approach is a strength, it also introduces challenges. The amalgamation of insights from multiple disciplines means that not all nuances of each discipline may have been explored in-depth, potentially leaving some aspects superficially addressed.

Future Research Directions: Expand Data Horizons: Future research could focus on aggregating and analyzing a more diverse dataset, possibly incorporating real-time data streams from ports across different global regions. This could help refine machine learning models further and enhance their global applicability.

Hybrid Models: There's potential in exploring hybrid machine learning models, combining strengths of different algorithms to offset individual limitations and ensuring faster and more accurate decision-making.

Operational Contexts: Extending the application of these methodologies to different operational contexts within the port logistics domain, such as warehousing, inventory management, or customs clearance, can be an intriguing avenue.

Environmental Impact Analysis: Given the increasing emphasis on sustainability, a more detailed exploration of the environmental impacts of optimized dispatching and port operations would be invaluable.

Technological Integration: With the emergence of technologies like the Internet of Things (Internet of Things (IoT)) and 5G connectivity, future research could further explore how these can be integrated with machine learning methodologies to enhance real-time data collection and decision-making in port operations.

Human Factor Analysis: It would be enlightening to assess the impact of these automated and optimized dispatching strategies on the human workforce in ports regarding labor dynamics and training requirements.

Significance Analysis: In the experiments, the t-test was used for significance testing without analyzing whether the results conform to a normal distribution. The effectiveness of the t-test may be compromised with data that do not follow a normal distribution. Verifying the normality of the data before selecting the appropriate method could enhance the persuasiveness.

In summation, while this thesis has made significant strides in the domain of container port logistics, the journey is by no means concluded. The limitations acknowledged provide a grounding perspective, ensuring that the research remains connected to practical realities. Simultaneously, the outlined future research directions illuminate the path forward, ensuring that the momentum achieved herein propels the field toward further innovation and excellence.

Bibliography

Abdelmagid, A. M., Gheith, M. S., and Eltawil, A. B. (2022). A comprehensive review of the truck appointment scheduling models and directions for future research. *Transport reviews*, 42(1):102–126.

Ahvanooey, M. T., Li, Q., Wu, M., and Wang, S. (2019). A survey of genetic programming and its applications. *KSII Transactions on Internet and Information Systems (TIIS)*, 13(4):1765–1794.

Arango, C., Cortés, P., Onieva, L., and Escudero, A. (2013). Simulation-optimization models for the dynamic berth allocation problem. *Computer-Aided Civil and Infrastructure Engineering*, 28(10):769–779.

Ardeh, M. A., Mei, Y., Zhang, M., and Yao, X. (2022). Knowledge transfer genetic programming with auxiliary population for solving uncertain capacitated arc routing problem. *IEEE Transactions on Evolutionary Computation*.

Arvin, R., Khattak, A. J., Kamrani, M., and Rio-Torres, J. (2020). Safety evaluation of connected and automated vehicles in mixed traffic with conventional vehicles at intersections. *Journal of Intelligent Transportation Systems*, 25(2):170–187.

Bai, R., Burke, E. K., Kendall, G., and McCullum, B. (2006). A simulated annealing hyper-heuristic for university course timetabling problem. *Abstract) PATAT*, 6.

Behdani, B., Wiegmans, B., Roso, V., and Haralambides, H. (2020). Port-hinterland transport and logistics: emerging trends and frontier research. *Maritime Economics & Logistics*, 22:1–25.

Bergqvist, R. (2012). 13 hinterland logistics and global supply chains. *Maritime logistics: a complete guide to effective shipping and port management*, page 211.

Berkes, F., Hughes, T. P., Steneck, R. S., Wilson, J. A., Bellwood, D. R., Crona, B., Folke, C., Gunderson, L., Leslie, H., Norberg, J., et al. (2006). Globalization, roving bandits, and marine resources. *Science*, 311(5767):1557–1558.

Bi, Y., Xue, B., and Zhang, M. (2020). Genetic programming with a new representation to automatically learn features and evolve ensembles for image classification. *IEEE transactions on cybernetics*, 51(4):1769–1783.

Bish, E. K., Chen, F. Y., Leong, Y. T., Nelson, B. L., Ng, J. W. C., and Simchi-Levi, D. (2007). Dispatching vehicles in a mega container terminal. In *Container terminals and cargo systems*, pages 179–194. Springer.

Brameier, M. F. and Banzhaf, W. (2007). *Linear genetic programming*. Springer Science & Business Media.

Burke, E., Kendall, G., Newall, J., Hart, E., Ross, P., and Schulenburg, S. (2003). Hyper-heuristics: An emerging direction in modern search technology. *Handbook of metaheuristics*, pages 457–474.

Burke, E. K., Gendreau, M., Hyde, M., Kendall, G., Ochoa, G., Özcan, E., and Qu, R. (2013). Hyper-heuristics: A survey of the state of the art. *Journal of the Operational Research Society*, 64:1695–1724.

Burke, E. K., Hyde, M., Kendall, G., Ochoa, G., Özcan, E., and Woodward, J. R. (2010a). A classification of hyper-heuristic approaches. In *Handbook of metaheuristics*, pages 449–468. Springer.

Burke, E. K., Hyde, M., Kendall, G., and Woodward, J. (2010b). A genetic programming hyper-heuristic approach for evolving 2-d strip packing heuristics. *IEEE Transactions on Evolutionary Computation*, 14(6):942–958.

Chao, S.-L. and Lin, Y.-L. (2017). Gate automation system evaluation: a case of a container number recognition system in port terminals. *Maritime Business Review*, 2(1):21–35.

Chen, B., Qu, R., Bai, R., and Laesanklang, W. (2018a). A hyper-heuristic with two guidance indicators for bi-objective mixed-shift vehicle routing problem with time windows. *Applied Intelligence*, 48(12):4937–4959.

Chen, C., Hsu, W.-J., and Huang, S.-Y. (2003). Simulation and optimization of container yard operations: A survey. In *Proceedings of international conference on port and maritime R and D and technology*, pages 23–29.

Chen, H., Wang, Y., Guo, T., Xu, C., Deng, Y., Liu, Z., Ma, S., Xu, C., Xu, C., and Gao, W. (2021). Pre-trained image processing transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12299–12310.

Chen, J., Bai, R., Dong, H., Qu, R., and Kendall, G. (2016). A dynamic truck dispatching problem in marine container terminal. In *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–8. IEEE.

Chen, L., Bostel, N., Dejax, P., Cai, J., and Xi, L. (2007). A tabu search algorithm for the integrated scheduling problem of container handling systems in a maritime terminal. *European Journal of Operational Research*, 181(1):40–58.

Chen, P., Fu, Z., Lim, A., and Rodrigues, B. (2004). Port yard storage optimization. *IEEE Transactions on Automation Science and Engineering*, 1(1):26–37.

Chen, Q., Xue, B., and Zhang, M. (2018b). Improving generalization of genetic programming for symbolic regression with angle-driven geometric semantic operators. *IEEE Transactions on Evolutionary Computation*, 23(3):488–502.

Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794.

Chen, X., Bai, R., and Dong, H. (2019). A multi-layer gp hyper-heuristic for real-time truck dispatching at a marine container terminal. *MISTA 2019*.

Chen, X., Bai, R., Qu, R., and Dong, H. (2022). Cooperative double-layer genetic programming hyper-heuristic for online container terminal truck dispatching. *IEEE Transactions on Evolutionary Computation*.

Chen, X., Bai, R., Qu, R., Dong, H., and Chen, J. (2020). A data-driven genetic programming heuristic for real-world dynamic seaport container terminal truck dispatching. In *2020 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8. IEEE.

Chen, X., Feiyang, B., Qu, R., Jing, D., and Bai, R. (2023). Neural network assisted genetic programming in dynamic container port truck dispatching. In *2023 IEEE International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–6. IEEE.

Chen, X., Zhang, H., Wu, C., Mao, S., Ji, Y., and Bennis, M. (2018c). Optimized computation offloading performance in virtual edge computing systems via deep reinforcement learning. *IEEE Internet of Things Journal*, 6(3):4005–4018.

Choi, H. R., Park, B. K., Lee, J., and Park, C. (2011). Dispatching of container trucks using genetic algorithm. In *The 4th International Conference on Interaction Sciences*, pages 146–151. IEEE.

Christodoulaki, E., Kampouridis, M., and Kanellopoulos, P. (2022). Technical and sentiment analysis in financial forecasting with genetic programming. In *2022 IEEE Symposium on Computational Intelligence for Financial Engineering and Economics (CIFEr)*, pages 1–8. IEEE.

Chu, J. C., Yan, S., and Chen, K.-L. (2012). Optimization of earth recycling and dump truck dispatching. *Computers & Industrial Engineering*, 62(1):108–118.

Clott, C. B. and Hartman, B. C. (2013). Clean trucks in california ports: modelling emissions policy. *International Journal of Shipping and Transport Logistics*, 5(4-5):449–462.

Cowling, P., Kendall, G., and Soubeiga, E. (2002). Hyperheuristics: A tool for rapid prototyping in scheduling and optimisation. In *Workshops on Applications of Evolutionary Computation*, pages 1–10. Springer.

Cui, M., Bai, R., Lu, Z., Li, X., Aickelin, U., and Ge, P. (2019). Regular expression based medical text classification using constructive heuristic approach. *IEEE Access*, 7:147892–147904.

Dai, J., Lin, W., Moorthy, R., and Teo, C.-P. (2008). Berth allocation planning optimization in container terminals. *Supply chain analysis: a handbook on the interaction of information, system and optimization*, pages 69–104.

de la Peña Zarzuelo, I., Soeane, M. J. F., and Bermúdez, B. L. (2020). Industry 4.0 in the port and maritime industry: A literature review. *Journal of Industrial Information Integration*, 20:100173.

de Santiago Junior, V. A., Özcan, E., and de Carvalho, V. R. (2020). Hyper-heuristics based on reinforcement learning, balanced heuristic selection and group decision acceptance. *Applied Soft Computing*, 97:106760.

Domović, D., Rolich, T., and Golub, M. (2019). Evolutionary hyper-heuristic for solving the strip-packing problem. *The Journal of The Textile Institute*, 110(8):1141–1151.

Drake, J. H., Kheiri, A., Özcan, E., and Burke, E. K. (2020). Recent advances in selection hyper-heuristics. *European Journal of Operational Research*, 285(2):405–428.

Ebner, M. (1999). On the search space of genetic programming and its relation to nature's search space. In *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)*, volume 2, pages 1357–1361. IEEE.

Ecer, F., Ardabili, S., Band, S. S., and Mosavi, A. (2020). Training multi-layer perceptron with genetic algorithms and particle swarm optimization for modeling stock price index prediction. *Entropy*, 22(11):1239.

Elhenawy, M., Chen, H., and Rakha, H. A. (2014). Dynamic travel time prediction using data clustering and genetic programming. *Transportation Research Part C: Emerging Technologies*, 42:82–98.

Elshaikh, A., Salhi, S., Brimberg, J., Mladenović, N., Callaghan, B., and Nagy, G. (2016). An adaptive perturbation-based heuristic: An application to the continuous p-centre problem. *Computers & Operations Research*, 75:1–11.

Emuna, R., Borowsky, A., and Biess, A. (2020). Deep reinforcement learning for human-like driving policies in collision avoidance tasks of self-driving cars. *arXiv preprint arXiv:2006.04218*.

Fahdi, S., Elkhechafi, M., and Hachimi, H. (2021). Machine learning for cleaner production in port of casablanca. *Journal of Cleaner Production*, 294:126269.

Fan, Q., Bi, Y., Xue, B., and Zhang, M. (2022). Genetic programming for image classification: a new program representation with flexible feature reuse. *IEEE Transactions on Evolutionary Computation*.

Fan, Z., Wang, Z., Li, W., Zhu, X., Hu, B., Zou, A.-M., Bao, W., Gu, M., Hao, Z., and Jin, Y. (2023). Automated pattern generation for swarm robots using constrained multi-objective genetic programming. *Swarm and Evolutionary Computation*, page 101337.

Filom, S., Amiri, A. M., and Razavi, S. (2022). Applications of machine learning methods in port operations–a systematic literature review. *Transportation Research Part E: Logistics and Transportation Review*, 161:102722.

Fogel, L. J., Owens, A. J., and Walsh, M. J. (1966). *Artificial intelligence through simulated evolution*. Wiley.

Garrido, P. and Riff, M. C. (2010). Dvrp: a hard dynamic combinatorial optimisation problem tackled by an evolutionary hyper-heuristic. *Journal of Heuristics*, 16:795–834.

Golias, M. M., Saharidis, G. K., Boile, M., Theofanis, S., and Ierapetritou, M. G. (2009). The berth allocation problem: Optimizing vessel arrival time. *Maritime Economics & Logistics*, 11:358–377.

Gould, H., Tobochnik, J., and Christian, W. (2007). An introduction to computer simulation methods. *Comput. Phys*, 10:652–653.

Hall, P. V. (2009). Container ports, local benefits and transportation worker earnings. *GeoJournal*, 74:67–83.

Harvey, H. B. and Sotardi, S. T. (2018). The pareto principle. *Journal of the American College of Radiology*, 15(6):931.

He, J., Tan, C., and Zhang, Y. (2019). Yard crane scheduling problem in a container terminal considering risk caused by uncertainty. *Advanced Engineering Informatics*, 39:14–24.

He, J., Zhang, W., Huang, Y., and Yan, W. (2013). A simulation optimization method for internal trucks sharing assignment among multiple container terminals. *Advanced Engineering Informatics*, 27(4):598–614.

Heilig, L., Stahlbock, R., and Voß, S. (2020). From digitalization to data-driven decision making in container terminals. *Handbook of terminal planning*, pages 125–154.

Hermansyah, D. and Muklason, A. (2020). Evaluation of hyper-heuristic method using random-hill climbing algorithm in the examination timetabling problem. In *Journal of Physics: Conference Series*, volume 1569, page 022101. IOP Publishing.

Hildebrandt, T. and Branke, J. (2015). On using surrogates with genetic programming. *Evolutionary computation*, 23(3):343–367.

Hoai, N. X., McKay, R. I., and Essam, D. (2006). Representation and structural difficulty in genetic programming. *IEEE Transactions on evolutionary computation*, 10(2):157–166.

Hong, L., Wang, G., Özcan, E., and Woodward, J. (2024). Ensemble strategy using particle swarm optimisation variant and enhanced local search capability. *Swarm and Evolutionary Computation*, 84:101452.

Hoos, H. H. and Stützle, T. (2004). *Stochastic local search: Foundations and applications*. Elsevier.

Hubbs, C. D., Perez, H. D., Sarwar, O., Sahinidis, N. V., Grossmann, I. E., and Wassick, J. M. (2020). Or-gym: A reinforcement learning library for operations research problems. *arXiv preprint arXiv:2008.06319*.

Indraratna, B., Rujikiatkamjorn, C., Ameratunga, J., and Boyle, P. (2011). Performance and prediction of vacuum combined surcharge consolidation at port of brisbane. *Journal of Geotechnical and Geoenvironmental Engineering*, 137(11):1009–1018.

Irannezhad, E., Prato, C. G., and Hickman, M. (2020). An intelligent decision support system prototype for hinterland port logistics. *Decision Support Systems*, 130:113227.

Jaoua, A., Gamache, M., and Riopel, D. (2012). Specification of an intelligent simulation-based real time control architecture: Application to truck control system. *Computers in Industry*, 63(9):882–894.

Juan, A. A., Faulin, J., Pérez-Bernabeu, E., and Domínguez, O. (2013). Simulation-optimization methods in vehicle routing problems: a literature review and an example. In *Modeling and Simulation in Engi-*

neering, Economics, and Management: International Conference, MS 2013, Castellón de la Plana, Spain, June 6-7, 2013. Proceedings, pages 115–124. Springer.

Kaveshgar, N. and Huynh, N. (2015). A genetic algorithm heuristic for solving the quay crane scheduling problem with time windows. *Maritime Economics & Logistics*, 17(4):515–537.

Keceli, Y. (2016). A simulation model for gate operations in multi-purpose cargo terminals. *Maritime Policy & Management*, 43(8):945–958.

Khayyam, H., Jamali, A., Assimi, H., and Jazar, R. N. (2020). Genetic programming approaches in design and optimization of mechanical engineering applications. *Nonlinear Approaches in Engineering Applications: Automotive Applications of Engineering Problems*, pages 367–402.

Khorasgani, H., Wang, H., and Gupta, C. (2020). Challenges of applying deep reinforcement learning in dynamic dispatching. *arXiv preprint arXiv:2011.05570.*

Kim, K. H. and Park, Y.-M. (2004). A crane scheduling method for port container terminals. *European Journal of operational research*, 156(3):752–768.

Klug, N., Chauhan, A., V, V., and Ragala, R. (2019). k-rnn: Extending nn-heuristics for the tsp. *Mobile Networks and Applications*, 24:1210–1213.

Koza, J. R. (1994). Genetic programming ii: Automatic discovery of reusable subprograms. *Cambridge, MA, USA*, 13(8):32.

Kumari, S. (2021). Interplay of ai-driven maritime logistics: An in-depth research into port management, advanced operations automation, and crm integration for optimized performance and efficiency. *ESP Journal of Engineering and Technology Advancements*, 1(1):1–5.

Kuncheva, L. I. (2014). *Combining pattern classifiers: methods and algorithms*. John Wiley & Sons.

Kuzmicz, K. A. and Pesch, E. (2019). Approaches to empty container repositioning problems in the context of eurasian intermodal transportation. *Omega*, 85:194–213.

Lai, K. and Lam, K. (1994). A study of container yard equipment allocation strategy in hong kong. *International Journal of Modelling and Simulation*, 14(3):134–135.

Lai, K. and Leung, J. (2000). Analysis of gate house operations in a container terminal. *International Journal of Modelling and Simulation*, 20(1):89–94.

Lee, K., Laskin, M., Srinivas, A., and Abbeel, P. (2021). Sunrise: A simple unified framework for ensemble learning in deep reinforcement learning. In *International Conference on Machine Learning*, pages 6131–6141. PMLR.

Li, K. X., Luo, M., and Yang, J. (2012). Container port systems in china and the usa: a comparative study. *Maritime Policy & Management*, 39(5):461–478.

Liu, J., Bai, R., Lu, Z., Ge, P., Aickelin, U., and Liu, D. (2020). Data-driven regular expressions evolution for medical text classification using genetic programming. In *2020 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8. IEEE.

Lu, H.-A. and Jeng, J.-Y. (2006). Modeling and solution for yard truck dispatch planning at container terminal. In *Operations Research Proceedings 2005*, pages 117–122. Springer.

Ma, X., Li, X., Zhang, Q., Tang, K., Liang, Z., Xie, W., and Zhu, Z. (2018). A survey on cooperative co-evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 23(3):421–441.

Maturana, J., Lardeux, F., and Saubion, F. (2010). Autonomous operator management for evolutionary algorithms. *Journal of Heuristics*, 16:881–909.

Mei, Y., Chen, Q., Lensen, A., Xue, B., and Zhang, M. (2022). Explainable artificial intelligence by genetic programming: A survey. *IEEE Transactions on Evolutionary Computation*.

Mi, W. and Liu, Y. (2022). Smart port and artificial intelligence. In *Smart Ports*, pages 81–98. Springer.

Mirzaei-Nasirabad, H., Mohtasham, M., Askari-Nasab, H., and Alizadeh, B. (2023). An optimization model for the real-time truck dispatching problem in open-pit mining operations. *Optimization and Engineering*, pages 1–25.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. (2013). Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*.

Montavon, G., Samek, W., and Müller, K.-R. (2018). Methods for interpreting and understanding deep neural networks. *Digital signal processing*, 73:1–15.

Moros-Daza, A., Amaya-Mier, R., and Paternina-Arboleda, C. (2020). Port community systems: A structured literature review. *Transportation Research Part A: Policy and Practice*, 133:27–46.

Moyano, J. M. and Ventura, S. (2022). Auto-adaptive grammar-guided genetic programming algorithm to build ensembles of multi-label classifiers. *Information Fusion*, 78:1–19.

Mundhenk, T. N., Landajuela, M., Glatt, R., Santiago, C. P., Faissol, D. M., and Petersen, B. K. (2021). Symbolic regression via neural-guided genetic programming population seeding. *arXiv preprint arXiv:2111.00053*.

Myles, A. J., Feudale, R. N., Liu, Y., Woody, N. A., and Brown, S. D. (2004). An introduction to decision tree modeling. *Journal of Chemometrics: A Journal of the Chemometrics Society*, 18(6):275–285.

Nguyen, S., Mei, Y., and Zhang, M. (2017). Genetic programming for production scheduling: a survey with a unified framework. *Complex & Intelligent Systems*, 3(1):41–66.

Nguyen, S., Zhang, M., Johnston, M., and Tan, K. C. (2012). A computational study of representations in genetic programming to evolve dispatching rules for the job shop scheduling problem. *IEEE Transactions on Evolutionary Computation*, 17(5):621–639.

Nguyen, S., Zhang, M., Johnston, M., and Tan, K. C. (2013). Automatic design of scheduling policies for dynamic multi-objective job shop scheduling via cooperative coevolution genetic programming. *IEEE Transactions on Evolutionary Computation*, 18(2):193–208.

Nievergelt, J. (2000). Exhaustive search, combinatorial optimization and enumeration: Exploring the potential of raw computing power. In *Sofsem*, pages 18–35. Springer.

Pardalos, P. and Romeijn, E. (2002). Handbook of global optimization-volume 2: Heuristic approaches.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830.

Perkis, T. (1994). Stack-based genetic programming. In *Proceedings of the First IEEE Conference on Evolutionary Computation. IEEE World Congress on Computational Intelligence*, pages 148–153. IEEE.

Pillay, N. and Qu, R. (2018). *Hyper-heuristics: theory and applications.* Springer.

Pluhacek, M., Senkerik, R., Viktorin, A., Kadavy, T., and Zelinka, I. (2018). A review of real-world applications of particle swarm optimization algorithm. *AETA 2017-Recent Advances in Electrical Engineering and Related Sciences: Theory and Application*, pages 115–122.

Polikar, R. (2006). Ensemble based systems in decision making. *IEEE Circuits and systems magazine*, 6(3):21–45.

Potter, M. A. and Jong, K. A. D. (2000). Cooperative coevolution: An architecture for evolving coadapted subcomponents. *Evolutionary computation*, 8(1):1–29.

Qin, S., Pi, D., Shao, Z., and Xu, Y. (2022). A cluster-based cooperative co-evolutionary algorithm for multiobjective workflow scheduling in a cloud environment. *IEEE Transactions on Automation Science and Engineering.*

Qin, W., Zhuang, Z., Huang, Z., and Huang, H. (2021). A novel reinforcement learning-based hyper-heuristic for heterogeneous vehicle routing problem. *Computers & Industrial Engineering*, 156:107252.

Rajendran, S. (2021). Real-time dispatching of air taxis in metropolitan cities using a hybrid simulation goal programming algorithm. *Expert Systems with Applications*, 178:115056.

Remde, S., Cowling, P., Dahal, K., Colledge, N., and Selensky, E. (2012). An empirical study of hyperheuristics for managing very large sets of low level heuristics. *Journal of the operational research society*, 63(3):392–405.

Rodrigue, J.-P. and Notteboom, T. (2009). The terminalization of supply chains: reassessing the role of terminals in port/hinterland logistical relationships. *Maritime Policy & Management*, 36(2):165–183.

Ryser-Welch, P. and Miller, J. F. (2014). A review of hyper-heuristic frameworks. In *Proceedings of the evo20 workshop, aisb*, volume 2014.

Sarmiento, M. G. C., Epprecht, E. K., Oliveira, F. L. C., Rodrigues, A. T. S., and Canchumuni, S. W. A. (2019). The use of simulation to model the dispatch of inbound containers in port terminals. *Pesquisa Operacional*, 39:155–175.

Schonfeld, P. and Sharafeldien, O. (1985). Optimal berth and crane combinations in containerports. *Journal of waterway, port, coastal, and ocean engineering*, 111(6):1060–1072.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Sha, M., Zhang, T., Lan, Y., Zhou, X., Qin, T., Yu, D., and Chen, K. (2017). Scheduling optimization of yard cranes with minimal energy consumption at container terminals. *Computers & Industrial Engineering*, 113:704–713.

Sislioglu, M., Celik, M., and Ozkaynak, S. (2019). A simulation model proposal to improve the productivity of container terminal operations through investment alternatives. *Maritime Policy & Management*, 46(2):156–177.

Sobania, D. (2021). On the generalizability of programs synthesized by grammar-guided genetic programming. In *European Conference on Genetic Programming (Part of EvoStar)*, pages 130–145. Springer.

Song, D. (2021). A literature review, container shipping supply chain: Planning problems and research opportunities. *Logistics*, 5(2):41.

Song, D.-P., Li, D., and Drake, P. (2015). Multi-objective optimization for planning liner shipping service with uncertain port times. *Transportation Research Part E: Logistics and Transportation Review*, 84:1–22.

Song, Y., Suganthan, P. N., Pedrycz, W., Ou, J., He, Y., Chen, Y., and Wu, Y. (2023). Ensemble reinforcement learning: A survey. *Applied Soft Computing*, page 110975.

Swan, J., Woodward, J., Özcan, E., Kendall, G., and Burke, E. (2014). Searching the hyper-heuristic design space. *Cognitive Computation*, 6:66–73.

Talley, W. K. (2006a). Chapter 22 port performance: An economics perspective. *Research in Transportation Economics*, 17:499–516. Devolution, Port Governance and Port Performance.

Talley, W. K. (2006b). Optimum port throughput. Technical report.

Tao, J. and Qiu, Y. (2015). A simulation optimization method for vehicles dispatching among multiple container terminals. *Expert systems with Applications*, 42(7):3742–3750.

Theodorakos, K., Agudelo, O. M., Schreurs, J., Suykens, J. A., and De Moor, B. (2022). Island transpeciation: A co-evolutionary neural architecture search, applied to country-scale air-quality forecasting. *IEEE Transactions on Evolutionary Computation*.

Ting, C.-J., Wu, K.-C., and Chou, H. (2014). Particle swarm optimization algorithm for the berth allocation problem. *Expert Systems with Applications*, 41(4):1543–1550.

Tsou, M.-C. (2019). Big data analysis of port state control ship detention database. *Journal of Marine Engineering & Technology*, 18(3):113–121.

Van Hasselt, H., Guez, A., and Silver, D. (2016). Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

Venturini, G., Iris, Ç., Kontovas, C. A., and Larsen, A. (2017). The multi-port berth allocation problem with speed optimization and emission considerations. *Transportation Research Part D: Transport and Environment*, 54:142–159.

Wang, T., Chen, J., Lü, J., Liu, K., Zhu, A., Snoussi, H., and Zhang, B. (2022). Synchronous spatiotemporal graph transformer: A new framework for traffic data prediction. *IEEE Transactions on Neural Networks and Learning Systems*.

Whigham, P. A. et al. (1995). Grammatically-based genetic programming.

In *Proceedings of the workshop on genetic programming: from theory to real-world applications*, volume 16, pages 33–41. Citeseer.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Scao, T. L., Gugger, S., Drame, M., Lhoest, Q., and Rush, A. M. (2020a). Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., et al. (2020b). Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45.

Wolpert, D. H. and Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, 1(1):67–82.

Wu, Y., Xiong, X., Gang, X., and Nyberg, T. R. (2013). Study on intelligent port under the construction of smart city. In *Proceedings of 2013 IEEE International Conference on Service Operations and Logistics, and Informatics*, pages 175–179. IEEE.

Xisong, D., Gang, X., Yuantao, L., Xiujiang, G., and Yisheng, L. (2013). Intelligent ports based on internet of things. In *Proceedings of 2013 IEEE International Conference on Service Operations and Logistics, and Informatics*, pages 292–296. IEEE.

Xu, Y., Fang, M., Chen, L., Xu, G., Du, Y., and Zhang, C. (2021). Reinforcement learning with multiple relational attention for solving vehicle routing problems. *IEEE Transactions on Cybernetics*, 52(10):11107–11120.

Yang, J. and Guo, L. (2020). Optimization of marine port logistics collection and distribution network: a perspective of supply chain management. *Journal of coastal research*, 106(SI):473–476.

Yi, W., Qu, R., Jiao, L., and Niu, B. (2022). Automated design of metaheuristics using reinforcement learning within a novel general search framework. *IEEE Transactions on Evolutionary Computation*.

Zhang, C., Liu, J., Wan, Y.-w., Murty, K. G., and Linn, R. J. (2003). Storage space allocation in container terminals. *Transportation Research Part B: Methodological*, 37(10):883–903.

Zhang, F., Mei, Y., Nguyen, S., and Zhang, M. (2023). Survey on genetic programming and machine learning techniques for heuristic design in job shop scheduling. *IEEE Transactions on Evolutionary Computation*.

Zhang, H., Zhou, A., and Zhang, H. (2021a). An evolutionary forest for regression. *IEEE Transactions on Evolutionary Computation*, 26(4):735–749.

Zhang, Y., Bai, R., Qu, R., Tu, C., and Jin, J. (2021b). A deep reinforcement learning based hyper-heuristic for combinatorial optimisation with uncertainties. *European Journal of Operational Research*.

Zhang, Y., Bai, R., Qu, R., Tu, C., and Jin, J. (2022). A deep reinforcement learning based hyper-heuristic for combinatorial optimisation with uncertainties. *European Journal of Operational Research*, 300(2):418–427.

Zhao, F., Di, S., Cao, J., Tang, J., et al. (2021). A novel cooperative multi-stage hyper-heuristic for combination optimization problems. *Complex System Modeling and Simulation*, 1(2):91–108.

Zhao, H., Yang, H., Wang, Y., Wang, D., and Su, R. (2020). Attention based graph bi-lstm networks for traffic forecasting. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–6. IEEE.

Zhen, L. (2016). Modeling of yard congestion and optimization of yard template in container ports. *Transportation Research Part B: Methodological*, 90:83–104.

Zhen, L., Jiang, X., Lee, L. H., and Chew, E. P. (2013). A review on yard management in container terminals. *Industrial Engineering & Management Systems*, 12(4):289–304.

Zhen, L., Lee, L. H., and Chew, E. P. (2011). A decision model for berth allocation under uncertainty. *European Journal of Operational Research*, 212(1):54–68.

Zheng, G., Zhang, F., Zheng, Z., Xiang, Y., Yuan, N. J., Xie, X., and Li, Z. (2018). Drn: A deep reinforcement learning framework for news recommendation. In *Proceedings of the 2018 world wide web conference*, pages 167–176.

Zhou, Z., Kearnes, S., Li, L., Zare, R. N., and Riley, P. (2019). Optimization of molecules via deep reinforcement learning. *Scientific reports*, 9(1):10752.