Multi Agent Reinforcement Learning Based Warehouse Task Assignment

K.A. Ashan Priyadarshana

218537D

MSc/PG Diploma in Artificial Intelligence

Department of Computational Mathematics

Faculty of Information Technology

University of Moratuwa

Sri Lanka

April 2024

Multi Agent Reinforcement Learning Based Warehouse Task Assignment

K.A. Ashan Priyadarshana

218537D

Thesis/Dissertation submitted in partial fulfillment of the requirements for the degree

MSc/PG Diploma in Artificial Intelligence

Department of Computational Mathematics

Faculty of Information Technology

University of Moratuwa

Sri Lanka

April 2024

DECLARATION

I declare that this is my own work and this thesis/dissertation does not incorporate without acknowledgement any material previously submitted for a degree or diploma in any other University or Institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text. I retain the right to use this content in whole or part in future works (such as articles or books).

Signature Date:

The above candidate has carried out research for the PhD/MPhil/Master's thesis/dissertation under my supervision. I confirm that the declaration made above by the student is true and correct.

Name of Supervisor: Prof. Asoka S. Karunananda

Signature of the Supervisor: Date:

DEDICATION

To my beloved parents, S.A Pathmawathi and K.A Asoka Priyantha. Your love, endless support, and unwavering belief in me have been the foundation of my life and my achievements. This work is a testament to your guidance and the importance you instilled in education.

ACKNOWLEDGEMENT

I would like to express my profound gratitude to my thesis supervisor, Professor Asoka S. Karunananda. From the very beginning of my master's program, his ability to inspire a fresh perspective has been transformative. His lectures were consistently engaging, and his guidance throughout my research was invaluable. I deeply appreciate his wisdom, support, and the positive impact he has had on my academic journey.

ABSTRACT

The growth of e-commerce and the demand for rapid fulfillment place a high pressure on modern warehouses to optimize efficiency. Central to this problem is the dynamic allocation of different tasks to a workforce comprised of both humans and robots. The task allocation system must prioritize time-sensitive picking tasks and ensure the timely completion of crucial supporting tasks like replenishment, all while considering the availability of specialized devices required for different task types. Conventional rule-based or heuristic systems struggle to adapt and optimize decision-making within such a complex operating environment.

Recent literature shows interest in applying reinforcement learning for warehouse management, including task allocation. Reinforcement Learning algorithms are well-suited to environments with clear rewards and observable states. However, most existing work focuses on single-agent scenarios or limited aspects of warehouse operations. In contrast, real-world warehouse task allocation inherently involves a cooperative multi-agent setting.

This thesis adopts a Multi-Agent Reinforcement Learning (MARL) framework to address the challenge of optimizing task assignment within a warehouse environment characterized by a workforce comprised of both human workers and autonomous robots. A central challenge addressed in this framework is the need to prioritize and allocate tasks with varying levels of urgency while considering the distinct constraints and capabilities associated with different worker types. To capture the dynamic nature and interdependencies of warehouse tasks, a comprehensive representation is used. Each task is defined by a set of attributes including its type (e.g., 'pick', 'replenish', 'load', or 'put'), the specific product or Stock Keeping Unit (SKU) involved, quantity, source and destination locations, timestamp, and order association. Notably, pick tasks are directly linked to specific customer orders for prioritization, while replenishment tasks may be indirectly related if they serve to stock locations for picking operations. Additionally, a 'status' attribute indicates whether the task is 'available', 'active', or 'completed'. Devices essential for task completion are modeled with attributes indicating their type (e.g., forklifts and pallet jacks), availability status, and any currently assigned tasks. The MARL system leverages agents that learn effective decision-making based on the current state of the warehouse. This state information is composed of two key elements provided as input to each agent's Deep Q-Network (DQN). The first element is the task state, which includes the current catalog of available tasks along with their detailed attributes. Secondly, device state information is fed into the DQN, providing details about the availability of devices and their current associations with tasks, if any. By considering both task information and device availability, the agents are equipped to make informed decisions for optimal task allocation.

A carefully designed reward mechanism is crucial to guide agents towards optimal task selection and resource utilization. The reward structure is designed to reinforce the timely completion of customer orders, prioritizing pick tasks. To ensure uninterrupted picking workflows, replenishment tasks necessary to facilitate pick operations receive a secondary level of reward. Overall warehouse efficiency is encouraged through a base-level reward for the completion of other tasks. Importantly, negative rewards are introduced as an obstacle against actions that lead to wasted

resources or unnecessary delays. This includes actions such as selecting tasks that are already in progress, attempting to use unavailable devices, or remaining idle.

At the core of the proposed framework is a Multi Agent Reinforcement Learning approach utilizing Deep Q-Networks (DQNs) as the decision-making policy for each agent. DQNs, a powerful form of reinforcement learning, allowing agents to learn the value of different action within a given state. Over time, as agents interact with the environment and observe the outcomes of their decisions, they learn to make optimal choices. The use of individual DQNs for each agent, both human and robotic, is critical. This individualized DQN approach allows for personalized learning plans tailored to the specific capabilities of agent type, making the overall task allocation system more effective and adaptable.

The framework is evaluated through a series of simulation experiments. Scenarios are designed to assess the MARL-based system against a baseline random policy, focusing on assessing the system's impact with a human-only workforce and limited devices and examining the system's performance and adaptability with the introduction of robots alongside human workers.

Simulation results demonstrate a consistent improvement in key efficiency metrics when more agents employ the learned DQN policy. Notably, pick task completion efficiency shows substantial improvements, indicating the framework's success in prioritizing time-critical outbound tasks. Furthermore, the MARL system demonstrates effective device utilization, with a decrease in idle resources. This thesis confirms MARL's potential for optimizing task allocation in warehouse environments with heterogeneous workforces. Future extensions could explore more sophisticated coordination techniques across agents and incorporate warehouse layout into the decision-making process.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Prolegomena

In the modern e-commerce landscape, warehouses serve as critical hubs in the supply chain, tasked with the rapid and efficient processing and distribution of goods. The introduction of cutting-edge technologies has introduced the possibility of integrating both human labor and robotic assistance to optimize these operations [1]. Multi-Agent Reinforcement Learning (MARL) emerges as a transformative approach to managing such dynamic environments, promising to enhance task allocation processes significantly [9]. This thesis explores the utilization of MARL to assign various warehouse tasks—particularly focusing on crucial pick and replenishment tasks—efficiently by balancing the workload among human and robotic agents and ensuring optimal use of devices like forklifts and pallet jacks.

The approach adopted in this research involves using a MARL framework where each agent (either a human worker or a robot) operates within a Deep Q-Network (DQN). This setup allows agents to learn and adapt to the warehouse environment dynamically, aiming to improve the efficiency of task assignments. The primary focus is on the strategic allocation of tasks based on their urgency and the availability of necessary devices, without the need to navigate or avoid collisions within the warehouse space.

1.2 Objectives

The primary aim of this research is to develop a robust technological solution, leveraging MARL, to enhance warehouse task assignment. The specific objectives include:

- A critical review of the existing warehouse process and task assignment strategies.
- A review of technologies used for warehouse task assignment strategies.
- To innovate a MARL-based system that ensures efficient task distribution and device utilization.
- To evaluate the performance of the system through Key Performance Indicators (KPIs) like task completion efficiency and the balance of task distribution among agents.

1.3 Background and Motivation

Warehouses are vital in today's fast-paced world, especially with online shopping becoming more popular. They need to work fast and smart, sending out orders and keeping track of stock [24]. With new tech like the Internet of Things (IoT) and Artificial Intelligence (AI), warehouses are getting better at deciding which tasks to do first. These tools help make sure important jobs like picking orders and restocking shelves are done right away [12].

But there's still room to get better. Even with robots and computers, some things are not done as well as they could be. The systems we have often work on their own, not together. This means we're not using all the help from robots and people as best we could. Recent researches that have been carried out shows that we could use multi-agent systems for this problem. This would let robots and people work together more closely, making everything run smoother.

Researchers have tried using computer simulations to figure out how to give jobs to workers. These simulations look at what needs to be done each day and try to change who does what [2][17]. They help make sure that both the things coming into the warehouse and the orders going out are dealt with better. But these tests usually look at workers one by one, not how they can work together. Multi-agent reinforcement learning (MARL), where intelligent agents learn through interaction and feedback, could take simulations to a new level by allowing both humans and robots within the environment to dynamically improve task allocation.

Some scientists have made systems that lets many agents—like stores and warehouses—work together [6]. They each look after their own stock and order more when needed. This is good because it could help make sure tasks in a warehouse are shared out in the best way. But when tested, this was only in a simple setting, not a real, busy warehouse. They also didn't check if their system could handle lots of different products.

Studies show that robots and tech are helping a lot, especially with so many people buying things online. There are robots that can move things around a warehouse on their own and systems that can get items from the shelves without needing a person [3]. But putting these new gadgets into old warehouses can be hard and expensive. And while robots are great, they sometimes don't work well with people. We need cheaper, smarter ways for them to work together.

Researchers also have tried using robots that can think for themselves to move things around more efficiently. They've made robots that know the best routes to take, saving time [13]. But they haven't really looked at how these robots work alongside the warehouse workers.

Other experts have looked at how robots can work together. For example, when moving lots of items at once, it might be quicker if robots work as a team [9]. But these ideas have only been tried in simple tests, not real, complicated warehouses. Plus, they didn't think about how robots and people should work together.

Scientific reports talk about how robots are changing the whole process of getting things from one place to another. They're doing jobs like sorting out stock and moving things around. This shows that robots could really help in warehouses too [14]. But again, making these robots work well in different kinds of warehouses, and with the people there, can be tough. We need new ideas to make it easier for robots and people to work as a team.

Resent studies have carried out to create smart ways to decide which jobs should be done by robots, using things like how fast a robot is or how much it can carry. This is super important for keeping a warehouse running smoothly [15]. But we need to see if these ideas can work when there are also human workers.

While work on MARL for warehouse tasks exists, there are still unexplored areas. Studies often focus on simple, controlled environments that don't reflect the complexity of a real warehouse. The scalability of these systems, particularly when handling diverse inventories and a hybrid human-robot workforce, needs further exploration. This thesis aims to address these gaps. The research will investigate how MARL can be successfully applied in large, realistic warehouse simulations. The focus will be on the complex interplay between human workers and various robotic systems to achieve optimized task allocation and improved efficiency.

1.4 Problem in Brief

Current warehouse task allocation systems often prioritize tasks for individual agents (humans and robots) in isolation. This approach misses potential gains in efficiency and adaptability that could arise from collaborative task management. While research has explored the use of multi-agent reinforcement learning (MARL) in simplified warehouse environments, its applicability and effectiveness in large-scale, realistic simulations with diverse product inventories and a hybrid workforce remains largely untested. This thesis aims to address this gap by investigating how MARL can be leveraged to optimize task allocation within complex warehouse environments where both humans and robots work together, with a focus on prioritizing the timely completion of outbound tasks critical to order fulfillment

1.5 A Novel Approach to Warehouse Task Assignment

The core approach is grounded in leveraging Multi-Agent Reinforcement Learning (MARL) to optimize warehouse task allocation. Agents, driven by individual Deep Q-Networks (DQNs), are trained to make autonomous decisions about task selection, prioritizing crucial picking and replenishment tasks for operational efficiency. This integration of MARL within a heterogeneous workforce of humans and robots aims to create a dynamic and intelligent system that can adapt to the evolving demands of warehouse logistics.

In order to achieve this, the solution focus on a sophisticated input-output process informed by real-time data on task availability and device status. Agents use this data to make informed decisions within a simulated warehouse environment. The decisions are dictated by a DQN-informed reward system that promotes the prioritization of tasks critical to customer satisfaction and warehouse efficiency. The success of these decisions is measured by KPIs, offering a quantitative assessment of the system's impact on warehouse operations. This methodological approach not only underpins the agent's learning curve but also provides warehouse administrators with actionable insights, ensuring continuous improvement in task distribution and device utilization.

The success of the MARL system can be clearly seen by the outputs it produces. The system is designed to make sure tasks are given out in the best way. It focuses on getting tasks done quickly and well, especially the tasks that involve sending items out of the warehouse. I check how well the system is working by looking at different Key Performance Indicators (KPIs) that the system outputs at the end of each simulation run. One KPI looks at how fast pick tasks are done. These

are important because they're connected to customer orders. Another KPI checks if all the workers have a fair amount of work. It also calculates a KPI for checking how often the workers and the tools they use are just waiting around.

1.6 Resource requirements

The implementation of the proposed MARL-based system requires both hardware and software resources:

- Hardware: A machine for running simulations and data processing.
- Software: Development tools including Python and libraries like TensorFlow for building and training the DQN models, and a simulation tool to build test and validate the proposed system (OpenAI gymnasium).

1.7 Structure of the Thesis

This thesis is organized into several chapters. Chapter 1 Introduces the research topic and methodology. The next chapter, Chapter 2 Reviews current literature and existing technologies relevant to warehouse task management. Chapter 3 discuss the theoretical underpinnings of MARL and its application in task allocation. Chapter 4 Describes the specific approach used in applying MARL for warehouse task assignment. Chapter 5 explains the system design and the interaction of different modules within the proposed framework. Chapter 6 discuss the implementation of the system. Chapter 7 presents the evaluation methodology, scenarios, and the effectiveness of the system based on the conducted simulations. Chapter 8 concludes the thesis with a summary of findings, limitations, and potential areas for future research.

1.8 Summary

This introductory chapter has outlined the critical considerations and challenges in warehouse task allocation, particularly in environments with both human and robotic agents. Existing approaches often treat task assignments in isolation, missing opportunities to optimize efficiency through collaboration. While MARL shows promise in simplified warehouse simulations, its efficacy in complex, realistic settings remain largely unexplored. This thesis proposes an innovative MARL-based solution for enhanced task distribution, focusing on the timely completion of outbound tasks and a balanced workload across the hybrid workforce. The system learns through Deep Q-Networks (DQNs), ensuring adaptability while prioritizing crucial picking and replenishment tasks. Evaluation of the system will involve simulations and the use of KPIs to determine its impact on warehouse operations.

CHAPTER 2

DEVELOPMENTS AND CHALLENGES IN WAREHOUSE TAKS ASSIGNMENT

2.1 Introduction

In Chapter 1, I have presented the overall picture of this thesis covering the research problem and the essentials of the solution. This chapter gives a comprehensive literature review around warehouse task assignment. The literature review has been structured to cover the gestation of warehouse operations, task assignment techniques, their developments, and future directions. Here I have also summarized the research challenges in warehouse task allocation and defined my research problem.

2.2 Technological Advancements in Warehouse Management

The integration of Industry 4.0 technologies such as the Internet of Things (IoT), Artificial Intelligence (AI), and autonomous vehicles has revolutionized task allocation within smart warehouses [1]. These technologies facilitate enhanced decision-making processes, optimizing task assignments based not only on urgency but also on logistical efficiency. For example, the use of AI enables predictive analytics for task prioritization, ensuring high-priority tasks such as picking and replenishment are handled promptly to maintain operational flow.

2.3 Impact on Task Allocation and Gaps

Significantly, the research [1] identifies the pivotal role of automation and real-time data in improving task allocation. Automated guided vehicles (AGVs) and robotic systems are employed to execute tasks autonomously, reducing reliance on human intervention and minimizing errors. Moreover, real-time data captured from IoT devices ensures continuous updating of task priorities based on dynamic warehouse conditions, thus supporting adaptive task management. Despite these advancements, Tubis and Rohman (2023) highlight existing gaps such as the underutilization of multi-agent systems that can dynamically collaborate to optimize task allocation further. The current implementations predominantly focus on independent systems where inter-agent collaboration is minimal. This observation aligns with the research problem addressed in this thesis, suggesting a significant opportunity for exploring how multi-agent systems can enhance collaborative task management in warehouses that employ both robots and human workers.

2.4 Task Assignment as an Optimization Problem

There are some recent researches focused on optimizing warehouse task assignments by treating it as an Optimization Problem. This involves using mathematical models and algorithms to determine the most effective task assignments. These methods consider various factors, such as minimizing travel distances, balancing workloads, and meeting time constraints.

Zhang and Wan proposed a two-stage method using genetic algorithms and the A* algorithm to assign and plan paths for different types of Automated Guided Vehicles (AGVs). This approach significantly reduced the number of AGVs needed and improved operational efficiency by minimizing task completion time and energy consumption [36]. Similarly, Kulak et al. integrated task scheduling and multi-agent path planning using cluster-based tabu search algorithms. Their method addressed both task scheduling and pathfinding issues, showing that a combined approach improves task completion times and reduces conflicts between agents [37].

Lesch et al. provided a comprehensive study on various optimization algorithms, evaluating their performance based on Pareto front quality metrics. They highlighted the effectiveness of multi-objective optimization in achieving balanced task assignments in warehouses [38]. Adil applied simulated annealing to optimize storage locations, which significantly improved order picking efficiency by minimizing travel distances [39]. Marchetti et al. developed a model for autonomous vehicle systems in warehouses, offering insights into optimal system configurations to maximize throughput and efficiency [40]. These studies collectively demonstrate the potential of optimization techniques in enhancing warehouse operations.

2.5 Simulation-Based Optimization for Worker Assignment

Ganbold et al. developed a novel simulation-based optimization method using a discrete-event simulation (O2DES) framework combined with a random neighborhood search method to address the problem of worker allocation in warehouses [2]. This approach enabled the adaptation of worker assignment in real-time based on daily workload variations, showcasing a potential increase in both inbound and outbound service levels with minimal adjustments to worker allocation.

While Ganbold et al. successfully demonstrates the application of simulation-based optimization to improve service levels, the study primarily focuses on single-agent scenarios where each worker is considered independently. The research highlights a gap in the exploration of multi-agent systems where multiple workers (agents) could collaborate or compete to optimize task allocation dynamically. This gap underscores the potential for integrating multi-agent reinforcement learning systems to further enhance operational efficiency in warehouse environments, particularly in scenarios involving both robots and human workers.

Khirwar et al. developed a multi-agent reinforcement learning (MARL) system for inventory management in a simulated supply chain environment [9]. Their approach involves multiple agents at stores and a warehouse, each managing their own inventory and placing replenishment orders. This system is particularly relevant as it addresses the optimization of task allocation in a dynamic environment where both humans and robots could function as these agents. The model focuses on cooperation among agents to meet overall supply chain goals, which aligns with the interest in task allocation to ensure important warehouse tasks are efficiently managed.

However, there are notable gaps and limitations in their approach. While the system designed by Khirwar et al. shows improvements over traditional methods, it has only been validated in a controlled simple environment. The real-world applicability of their MARL system remains

uncertain, especially under complex and unpredictable conditions typical in large-scale warehouses. Additionally, their study covers only a limited number of products and does not fully explore the scalability to handle diverse items typically found in modern warehouses. These limitations suggest a need for further research to adapt and test their system in more varied and realistic settings.

Dhaliwal provided a detailed examination of the integration of automation and robotics within warehouse management, focusing particularly on their adoption due to the surge in e-commerce demands [12]. The study reviews various technological innovations including Automated Storage and Retrieval Systems (AS/RS), Goods-to-Person (G2P) technology, and various forms of automated guided vehicles (AGVs). This comprehensive overview demonstrates how these technologies have been used to streamline operations and enhance the efficiency of task allocations in warehouses, pertinent to the optimization of multi-agent systems for task allocation in dynamic environments like modern warehouses.

However, Dhaliwal also notes critical limitations in the adoption of these technologies. While automation significantly improves efficiency, the real-world application often struggles with integration challenges, particularly in older warehouse setups not originally designed for modern robotic systems. Additionally, the research highlights the high initial costs and the complexity of maintaining such advanced technologies. These challenges underscore the need for more adaptable, cost-effective solutions in automation technology to better cater to the dynamic nature of warehouse environments, thus presenting a gap in research that future advancements in multi-agent systems could potentially address.

Vivaldini et al. developed and tested robotic forklifts within an intelligent warehouse environment, focusing on optimizing routing, path planning, and auto-localization [13]. The research demonstrates the application of an Automated Guided Vehicle (AGV) behavior determined by a routing algorithm, which computes the overall task execution time and the minimum global path of each AGV using a topological map. This study addresses the efficiency of task allocations through automation, aligning with the aim of optimizing task assignment in warehouses that utilize both human and robotic workers.

Despite the advancements shown by Vivaldini et al. their study presents a limitation that the study did not address the integration of these systems with human workers, which is crucial for environments where both human and robotic agents are employed.

Pinkam et al. investigated robot collaboration strategies within a warehouse environment using Automated Guided Vehicles (AGVs) to handle multi-item transportation tasks [14]. Their research focused on comparing individual and collaborative collection methods using simulations in MATLAB, demonstrating how different strategies influence operational efficiency. The study is relevant for optimizing multi-agent systems in warehousing, particularly in scenarios where coordination between robots can significantly reduce the overall task completion time, a key factor for warehouses prioritizing swift task allocation and completion.

The study by Pinkam et al. however, has several limitations. Firstly, their simulations were confined to idealized, controlled settings which may not accurately mimic the complexities and

unpredictable dynamics of real-world warehouse operations. Additionally, their work did not address interactions between robots and human workers, an important aspect of modern warehouses that utilize a hybrid workforce.

Rajana discusses the incorporation of robotics in the supply chain, focusing on the role of automated systems in enhancing operational efficiency within logistics and e-commerce sectors [15]. The report covers the technological evolution and the integration of robotics into various business practices, illustrating how robotics has transformed supply chain management by automating tasks such as inventory handling and goods transportation. This exploration is particularly relevant for the development of multi-agent reinforcement learning systems aimed at optimizing task allocation in warehouses, as it underscores the potential of robotics to streamline operations and improve task execution speed in complex environment.

However, the implementation of robotics as outlined by Rajana is not without challenges. The report notes significant gaps such as the high costs associated with integrating sophisticated robotic systems and the technological limitations in adapting these systems to diverse operational settings. Furthermore, the study emphasizes the need for further innovation in robotic technology to ensure seamless integration and interaction with human workers, highlighting an ongoing need for research into more flexible, cost-effective, and cooperative robotic solutions capable of operating effectively in the dynamic and varied conditions of modern warehouses [15]. These limitations indicate critical areas for future research, particularly in enhancing the adaptability and efficiency of multi-agent systems within the warehouse task allocation domain.

Oliveira et al. developed an efficient task allocation algorithm for smart warehouses utilizing multi-delivery stations and a heterogeneous fleet of robots [16]. Their work focuses on minimizing operational costs by calculating optimal task assignments based on variable characteristics of robots, such as speed and load capacity. This approach is directly relevant to modern warehousing environments where task efficiency and robot utilization are critical, aligning well with the goal of optimizing task allocation using multi-agent reinforcement learning systems.

2.6 Automated storage and retrieval systems (AS/RS)

In addressing the complexities of warehouse management, Geng et al. (2022) explore automated storage and retrieval systems (AS/RS), emphasizing the significant enhancement these systems bring to warehouse operations through scheduling optimization [3]. Their study introduces a novel approach employing a dual-stackers model to manage task allocation effectively, demonstrating the potential for advancements in multi-agent systems within warehouse environments. Geng et al. presents a new improved genetic algorithm (NIGA) that tailors the algorithm structure to the population fitness density, optimizing the path of stackers within the AS/RS. This approach not only increases efficiency but also adapts to varying daily workloads, showcasing a substantial improvement in task handling and operational throughput.

While Geng et al. significantly advance the scheduling strategy by integrating sophisticated algorithms and dual stacker systems, the study primarily focuses on algorithmic efficiency and collision avoidance. The research does not extend to the collaborative dynamics of multi-agent

systems involving both robots and human workers, nor does it address the prioritization of tasks based on urgency, a critical aspect of modern warehousing operations. This gap indicates a pivotal area for further research, particularly in applying multi-agent reinforcement learning to enhance decision-making processes and task allocation efficacy in real-time.

2.7 Multi Agent systems in real-world warehouses

Rolf et al. conducted a comprehensive review of reinforcement learning applications in supply chain management [11]. Their research categorizes different reinforcement learning algorithms and evaluates their use across various aspects of supply chain operations, including inventory management, which is the most frequently studied area. The significance of their work lies in demonstrating how reinforcement learning can adapt supply chain decision-making to dynamic environments, thus aligning closely with task allocation and operational optimization in warehouses. This provides a foundation for understanding how multi-agent systems could enhance efficiency in modern warehousing operations where various tasks need dynamic assignment.

Rolf et al. also identifies several limitations in the current research landscape [11]. Notably, the majority of studies utilize simulated, often oversimplified, environments with artificial data, which may not accurately represent the complexities of real-world supply chains. This gap highlights the need for further research into the application of reinforcement learning in real industrial settings, where factors such as unpredictability of human actions and physical constraints play a significant role. Addressing these challenges is crucial for the practical deployment of reinforcement learning in dynamic and complex warehouse environments, where the ability to adapt quickly to changes is essential.

Ács et al. developed a heuristic optimization approach to split orders into warehouse carts efficiently and assign these carts to human agents, minimizing the total wage cost and irritation of workers [4]. This was achieved through enhancements to Multi-Agent Path Finding (MAPF) solution techniques, demonstrating the practical application of multi-agent systems to streamline warehouse operations.

While the study advances the application of heuristic optimizations and MAPF, it primarily addresses the operational efficiency from a cost and conflict resolution perspective. There is an opportunity to further explore how multi-agent reinforcement learning could dynamically adapt to real-time operational changes, particularly in warehouses that use both robots and human workers in more integrated task assignments. This could potentially improve the responsiveness and efficiency of task allocation beyond what is achieved through static daily optimizations.

Burggraf et al. developed a novel approach by integrating policy gradient algorithms with actor-critic architectures in a multi-agent system context [5]. This method addresses the real-time adjustments required in scheduling by adapting to disruptions and changes in the manufacturing environment, thus ensuring high-quality scheduling solutions with quick response times.

While the study primarily focuses on job shop scheduling, the principles and methodologies can be extrapolated to warehouse task allocation, especially in how tasks are dynamically assigned to

various agents. However, the research does not specifically tackle the integration of humans and robots within the same system, nor does it focus on prioritizing tasks based on their urgency or importance.

The study done by Jiang and Sheng (2009) introduces a CRL algorithm that optimizes inventory control by learning from past cases to make informed decisions under nonstationary demand conditions [6]. This approach mirrors the potential application of reinforcement learning in warehouse environments where task allocation must respond to fluctuating demands and priorities. Although the study provides a framework for using reinforcement learning in supply chains, the direct applicability to warehouse task allocation is limited. Their focus on inventory levels rather than on the allocation of tasks to agents in a warehouse means that the methods need significant adaptation to be fully applicable for task assignment problem.

The study done by Krnjaic et al. introduces a hierarchical MARL framework where a manager agent assigns tasks to worker agents, aiming to optimize the global objective of maximizing order throughput [7]. This approach not only enhances sample efficiency but also significantly improves pick rates across various warehouse configurations compared to traditional heuristic methods. This use of MARL demonstrates its success on flexible and effective solution to the dynamic and complex environment of modern warehouses. The ability of MARL to adapt to different warehouse layouts and worker types offers a direct application to task assignment problem, providing a robust framework.

While Krnjaic et al. present a groundbreaking approach to task allocation in warehouses, the study focuses predominantly on the picker-to-pick paradigm and does not address the integration of human and robot workers beyond the order-picking scenario.

Schäfer (2022) explored the challenges of generalization in multi-agent reinforcement learning (MARL) with a focus on task allocation in environments like warehouses [8]. The study emphasized the limitations of traditional reinforcement learning approaches, which tend to be highly specialized and lack the flexibility needed for varied real-world tasks. Through a series of preliminary experiments, Schäfer demonstrated the effectiveness of various generalization strategies, including the use of different observation encodings, neural network architectures, and the potential of meta reinforcement learning. These experiments sought to improve how MARL systems can adapt their learned behaviors to new, similar tasks without the need for retraining from scratch, which is directly relevant to optimizing task allocation in warehouses that employ both robots and humans.

However, the research also pointed out some important gaps and limitations. The methods tested for helping the system adapt to new warehouse layouts, like zero-shot generalization and fine-tuning, didn't fully succeed. This suggests that these methods need more work before they can be effectively used. Since these approaches are still early in their development, they require further testing and improvement. This limitation is crucial because it affects the ability of these systems to adjust to different tasks in dynamic warehouse environments. Being unable to adapt effectively to varied environments limits the practical use of MARL for managing tasks in warehouses where flexibility is essential. This is a key area that needs more research and development.

Sultana et al. explored the application of multi-agent reinforcement learning (MARL) to manage inventory across a multi-node supply chain that includes a central warehouse and several retail stores [17]. This study is relevant to the problem of warehouse task allocation, particularly in environments where multiple products are managed simultaneously across various nodes, reflecting the complexity and dynamics of modern warehouses. Their novel MARL framework optimizes the distribution of multiple products from a warehouse to multiple stores, aiming to maximize sales and minimize wastage, a direct parallel to ensuring that critical tasks such as picking and replenishment are efficiently assigned to enhance warehouse throughput.

However, the study focuses exclusively on inventory distribution without considering the integration of human workers with robots, an element crucial to modern warehouses that employ a hybrid workforce.

2.8 Warehouse Management Benchmarks

Yang et al. developed a versatile Multi-Agent Reinforcement Learning (MARL) benchmark specifically tailored for inventory management, named MABIM [10]. This benchmark simulates a multi-echelon, multi-commodity inventory management system using the OpenAI Gym framework. The introduction of MABIM allows for comprehensive performance evaluations of both traditional operations research methods and contemporary MARL algorithms across a range of challenging scenarios, including scalability, cooperation, competition, generalization, and robustness. This development is crucial for enhancing task allocation strategies where both robots and humans operate.

2.9 Overview of Developments and challenges in the warehouse task assignment

This chapter on "Developments and Challenges in Warehouse Task Assignment" provided a thorough literature review on the evolution of task assignment methodologies in warehouse management, emphasizing the role of technological advancements and identifying existing gaps in current systems. Integration of Industry 4.0 technologies like the Internet of Things (IoT), Artificial Intelligence (AI), and autonomous vehicles has transformed task allocation within smart warehouses. These technologies enhance decision-making processes, allowing for optimized task assignments based on urgency and logistical efficiency. AI, for instance, enables predictive analytics that prioritize tasks to maintain operational flow, specifically targeting crucial tasks such as picking and replenishment.

Automated Guided Vehicles (AGVs) and robotic systems have greatly improved task allocation in warehouses by working independently, reducing mistakes and the need for human intervention. Real-time information from IoT devices is crucial for continually adjusting task priorities as warehouse conditions change, supporting a flexible approach to task management. However, research points out that there's still room to improve by using multi-agent systems to enable dynamic cooperation between agents.

Research by Ganbold et al. demonstrates simulation-based methods that can quickly adapt worker assignments based on daily work fluctuations, potentially improving service levels. Yet, these methods often treat agents as separate entities, missing the advantages of agents working together. Similarly, the work on multi-agent reinforcement learning (MARL) for inventory management looks promising for better task management in changing environments, but its effectiveness in complex, large warehouses need more study.

Dhaliwal discusses how the growth of e-commerce has pushed the integration of automation and robotics, focusing on technologies like Automated Storage and Retrieval Systems (AS/RS) and Goods-to-Person (G2P) technology. While these technologies have made operations more efficient, adapting them to older warehouses and the high costs of integration present significant challenges. Research by Vivaldini et al. and Pinkam et al. explores how to optimize task allocation with these automated systems but doesn't address how these technologies interact with human workers, which is vital in workplaces that use both humans and robots.

Additionally, research by Oliveira et al. offers algorithms for efficient task allocation that reduce costs by better using robots in smart warehouses. Meanwhile, Geng et al. improve AS/RS capabilities with new algorithms that optimize the movement of stackers, although their work does not cover how these systems work with human workers in multi-agent environments.

The need for research on multi-agent systems in real-world settings is underscored by studies such as those by Rolf et al., who highlight the use of reinforcement learning in supply chain management but acknowledge the gap in real-world applicability due to the simplified nature of simulations used in current studies. Bridging these gaps requires further exploration of how MARL and other multi-agent systems can be effectively implemented to adapt to the complex and unpredictable nature of modern warehouses, thereby improving the efficiency and adaptability of task allocation systems. This ongoing research aims to develop methodologies that not only enhance task efficiency but also integrate seamlessly with the dynamic interplay of human and robotic agents within contemporary warehouse environments.

2.10 Problem definition

In modern warehouse operations, the allocation of tasks to both human workers and robots is critical for enhancing efficiency and throughput. As e-commerce continues to grow, the complexity and volume of tasks within warehouses increase correspondingly, necessitating more sophisticated management solutions. Currently, most warehouse task assignment systems use technology that prioritizes tasks independently for each agent, whether human or robot. This approach often leads to inefficiencies due to a lack of coordination among the different agents working in the same environment.

3.11 Summary

This chapter has outlined my review of current warehouse task allocation strategies, highlighting both their strengths and shortcomings. Alongside this survey, I've developed the problem definition

for my project. The next chapter will discuss the technologies currently utilized in warehouses for task assignment. In it, I will also introduce Multi-Agent Reinforcement Learning as a promising technology for improving warehouse task assignment.

CHAPTER 3

THEORETICAL FOUNDATIONS OF REINFORCEMENT LEARNING AND MULTI AGENT SYSTEMS

3.1 Introduction

As the landscape of warehousing evolves with technological advancements, the integration of multi-agent reinforcement learning (MARL) emerges as a pivotal solution for enhancing task allocation in modern warehouses [1]. This approach utilizes both human and robotic agents to optimize the distribution of tasks such as picking and replenishing items, crucial for maintaining warehouse efficiency.

3.2 Simulation-Based Optimizations

In the study by Ganbold et al. (2020), a simulation-based optimization method was developed to allocate warehouse workers effectively, improving service levels while maintaining a constant workforce [2]. This method utilized the object-oriented discrete-event simulation (O2DES) framework and a random neighborhood search algorithm to integrate simulation outputs for decision-making. Such a method proves particularly effective in scenarios where worker skills and task precedence constraints complicate direct assignment solutions.

Relating this to the use of multi-agent reinforcement learning (MARL) for task allocation in a mixed workforce of humans and robots, several parallels and adaptations become evident. While Ganbold et al. focus on optimizing human worker assignments within given constraints, MARL could extend this by dynamically assigning tasks in real-time, considering not only human capabilities but also robotic efficiency. MARL's ability to learn and adapt to changing environments could enhance the simulation-based approach by continuously optimizing task allocations based on ongoing performance data, potentially increasing both inbound and outbound service levels more effectively than static simulation models. Thus, integrating MARL with simulation-based methods could provide a robust framework for managing complex, dynamic task allocation challenges in modern warehouses.

3.3 Genetic Algorithms for Warehouse Task Assignment

In the research by Geng et al. an advanced scheduling strategy for automated storage and retrieval systems (AS/RS) is presented, emphasizing the use of a new improved genetic algorithm (NIGA) to optimize the scheduling of dual stackers with anti-collision mechanisms [3]. This study critically adapts the scheduling model by integrating the operational variability and physical constraints of warehouse environments, significantly improving the efficiency of task execution in warehouses equipped with AS/RS.

Relating this approach to a multi-agent reinforcement learning (MARL) framework for task allocation where both humans and robots work cooperatively can offer substantial benefits. While Geng et al. focused on optimizing stacker movements to reduce collision and improve task throughput using genetic algorithms, MARL could dynamically assign and reassign tasks based on real-time operational efficiency and worker availability. Such integration could potentially lead to a more robust and adaptive system, where MARL not only optimizes task allocation based on current warehouse states but also learns and evolves to predict future task allocation needs, further enhancing the efficiency and responsiveness of the warehouse operations.

3.4 Multi-Agent Path Finding (MAPF) within Warehouses

In the study by Ács et al. a multi-agent system was optimized for real-world warehouse operations, focusing on human agents' task assignment to minimize wage costs and reduce operational conflicts [4]. They developed a heuristic for optimizing task sequences within warehouse carts, utilizing Multi-Agent Path Finding (MAPF) techniques to enhance efficiency and reduce worker irritation by intelligently sequencing task execution.

This methodology is particularly relevant for implementing multi-agent reinforcement learning (MARL) in warehouses where both humans and robots collaborate. Ács et al.'s approach, which effectively assigns tasks and sequences them to minimize conflict and streamline operations, could be adapted to include robots alongside humans. MARL could dynamically adjust task priorities and assignments in real-time based on ongoing operations, potentially enhancing the model by Ács et al. by integrating the learning aspect of MARL to continuously improve task allocation efficiency based on operational feedback.

3.5 Policy Gradients and Actor Critics for Warehouse Operations

Policy gradients and actor-critic methods are two key concepts in the domain of multi-agent reinforcement learning (MARL), which is a branch of machine learning where multiple agents learn to make decisions in an environment to maximize their cumulative reward.

In the study by Burggräf et al. a multi-agent-based deep reinforcement learning (MARL) approach was implemented for dynamic job scheduling in a manufacturing setting. The research showcased a method where each machine in the production line was treated as an agent. These agents utilized a policy gradient algorithm with an actor-critic architecture to make real-time scheduling decisions [5]. This system allowed for dynamic adjustments based on current production states, aiming to optimize the overall operational efficiency by reducing total lead times.

Policy Gradients: This technique directly adjusts the parameters of the policy—the strategy an agent follows to choose actions based on its observations—by computing gradients that maximize expected future rewards. In MARL, each agent can use policy gradients to independently update its policy, potentially leading to more cooperative or competitive strategies depending on the environment. The advantage of policy gradients is that they can handle high-dimensional action spaces and continuous action domains effectively.

Actor-Critic Methods: These methods split the task into two components: an actor, which proposes actions, and a critic, which evaluates actions taken by the actor by computing the value function (a measure of the goodness of a state or state-action combination). The critic's feedback helps the actor adjust its policy more effectively. In MARL, each agent can have its own actor and critic, allowing for complex interactions and learning dynamics among agents. The actor-critic framework is beneficial because it often converges faster than methods based solely on value functions or policies, as it combines the strengths of both approaches.

This method of using MARL for task scheduling in complex manufacturing environments can be highly applicable to modern warehouses that utilize a mix of human and robotic workers. By applying a similar MARL framework, warehouses can achieve a high level of task allocation efficiency. The technology enables continuous learning and adaptation to changing conditions, which is crucial for managing the diverse and time-sensitive tasks in warehouses. Therefore, the approach taken by Burggräf et al. could be effectively adapted to enhance task distribution among human and robotic agents, ensuring that critical tasks such as picking and replenishment are carried out promptly and efficiently [5].

3.6 Case-Based Reinforcement Learning

Jiang and Sheng developed a case-based reinforcement learning algorithm (CRL) for dynamic inventory control within a multi-agent supply-chain system [6]. This method focuses on learning the optimal inventory levels in environments where customer demand is nonstationary. Their approach enhances traditional inventory management methods by enabling adaptive responses to changing conditions, thereby reducing issues like overstocking or shortages.

In their implementation, Jiang and Sheng applied the CRL algorithm within a two-echelon supply chain model consisting of multiple customers and retailers. The CRL algorithm helped in determining the most effective times and quantities for restocking inventory, according to learned service level targets. By using this algorithm, they were able to dynamically adjust inventory strategies based on actual demand rather than relying on fixed reordering policies, showing an improvement in inventory control efficiency.

Critically, while this approach shows promise for dynamic inventory control, adapting it to a warehouse task assignment scenario involving both human and robotic workers might require further adjustments. Specifically, task allocation in a warehouse involves variables that go beyond inventory levels, such as worker availability, task urgency, and spatial considerations of task locations. Although the principles of reinforcement learning could be beneficial in learning and adapting to the operational environment of a warehouse, the specific model would need significant customization to address the direct interactions between humans and robots and the real-time allocation of diverse tasks.

3.7 Hierarchical MARL for Order Picking

Krnjaic et al. developed a scalable multi-agent reinforcement learning (MARL) approach for warehouse logistics. This approach involves hierarchical MARL algorithms where a manager assigns tasks to worker agents like robots and humans [7]. The aim is to improve cooperation among agents and increase efficiency in order-picking operations.

The technology was applied in a simulated environment with diverse warehouse configurations. This allowed for a flexible adaptation to various warehouse sizes, layouts, and types of workers. The hierarchical MARL algorithms demonstrated improved sample efficiency and pick rates compared to baseline MARL algorithms and traditional industry heuristics.

Critically, while this MARL approach shows promise for enhancing warehouse operations by enabling dynamic task allocation among both human and robot workers, its real-world application might need adjustments. The simulation environment cannot perfectly mimic all real-world complexities, such as the physical interactions between different agents and the unpredictability of human actions. Therefore, while the MARL approach could theoretically manage task allocation effectively, practical deployment would require extensive testing and possibly adaptations to handle real-world variability and ensure smooth human-robot cooperation.

3.8 Zero-Shot Task Generalization

Schäfer introduced a framework for task generalization in multi-agent reinforcement learning (MARL) applied to warehouse operations [8]. This approach utilizes the concept of zero-shot generalization and fine-tuning to adapt to varied warehouse environments without retraining from scratch. Schäfer's work emphasizes the importance of generalization in MARL, where agents need to perform well across diverse settings without specific prior training for each scenario.

In the context of reinforcement learning (RL) and multi-agent reinforcement learning (MARL), zero-shot generalization means that an agent or a group of agents can navigate and operate in an environment they have never encountered during their training phase. This is achieved by training the agents in a way that they develop a generalized understanding of their tasks that transcends the specifics of their training environments.

Fine-tuning, on the other hand, is a technique used to adapt a pre-trained model to a new but related task or to new but similar data. It involves taking a model that has been trained on a large dataset (often in a slightly different context) and continuing its training on a smaller, more specific dataset that is directly relevant to the task at hand. Fine-tuning could be used when a MARL system trained generally on multiple warehouse environments needs to be optimized for a specific new warehouse, enhancing the system's efficiency and effectiveness in that particular setting.

Critically, while Schäfer's approach offers significant advancements in generalization within MARL, applying this method to a mixed workforce of humans and robots in warehouse task assignment presents additional challenges [8]. Human-robot interactions introduce complexities such as varying speeds, efficiency, and the unpredictability of human behavior, which are not fully addressed by current MARL models

3.9 Cooperative Multi-Agent Reinforcement Learning for Efficient Inventory Management

The technology explored by Khirwar et al. is Cooperative Multi-Agent Reinforcement Learning (CMARL), tailored for inventory management in complex supply chains [9]. This system incorporates a novel architecture with an enhanced state and action space. It employs a shared reward system to optimize the entire supply chain's performance, focusing on cooperative interactions between multiple agents.

In their implementation, CMARL is deployed to control inventory flow between a central warehouse and multiple stores. Each node in the supply chain, represented by an agent, makes independent decisions based on local inventory levels to place replenishment orders upstream. The warehouse agent, aside from standard order placements, learns to allocate inventory efficiently to stores depending on their demands, balancing supply across the network.

Adapting CMARL to a mixed workforce of humans and robots in warehouse task assignments poses potential benefits and challenges. While CMARL's robust framework can enhance decision-making in dynamic environments, integrating human and robotic agents requires modifications to account for the diverse capabilities and coordination needs. The system's ability to learn and adapt dynamically could significantly improve task allocation efficiency, though real-world application would necessitate further development to fully integrate human factors and ensure effective human-robot collaboration.

3.10 Use of Heuristic Algorithms in Warehouses

The problem of task allocation in multi-robot systems (MRTA) within smart warehouses, characterized as NP-hard, generally necessitates heuristic solutions due to its complexity [16]. These algorithms compute task assignments based on robot characteristics such as traffic speed and load capacity, adapting to the dynamic and heterogeneous nature of robotic fleets. This methodology underscores the practical necessity of employing computational intelligence and IoT technologies to optimize logistical operations in warehouses increasingly reliant on automated systems.

Oliveira et al. present an innovative approach to task allocation that integrates a novel cost estimator [16]. This estimator functions by evaluating tasks based on the variable characteristics of robots, such as load capacity, to allocate warehouse tasks efficiently. By simulating various logistical scenarios, this strategy has demonstrated the potential to enhance route efficiency by up to 33%, significantly reducing operational costs compared to traditional methods. Such an approach is validated through rigorous testing in environments that closely mimic real-world smart warehouses with multiple delivery stations and heterogeneous robot fleets.

While the algorithm developed by Oliveira et al. (2022) proves highly effective in scenarios involving solely robotic agents, its adaptability to environments where humans and robots coexist remains to be rigorously evaluated. The algorithm's reliance on precise calculations of robot capacities and speeds may not directly translate to scenarios requiring human-robot collaboration,

where unpredictability and variability of human work patterns could introduce complexities not accounted for in the current model. Future adaptations could explore integrating human behavioral data into the cost estimator to better predict and optimize task allocation in mixed workforces.

3.11 Use of Multi-Agent Reinforcement Learning for Warehouses

The application of MARL in warehouse operations focuses on the strategic allocation of tasks without the necessity to manage the movement or avoid collisions among agents [17]. The practical application of MARL in warehousing is illustrated by its ability to handle complex, concurrent operations involving multiple tasks such as picking, replenishment, inbound, and loading processes. Each agent in the system is responsible for a subset of tasks and makes decisions to maximize overall operational efficiency. For example, in a scenario where tasks vary in priority and complexity, MARL can effectively allocate more resources to critical tasks such as picking, ensuring that high-priority orders are fulfilled swiftly and accurately. This adaptive decision-making process is crucial for modern warehouses that aim to reduce operational bottlenecks and improve response times.

This section delves into the specifics of how MARL can be adapted to improve the management of task assignments in warehouse settings:

Task Prioritization: MARL algorithms are adept at prioritizing tasks based on their importance, which is essential for time-sensitive operations like order picking. This ensures that high-priority tasks, such as picking items for pending orders, are completed promptly to meet delivery deadlines.

Dynamic Task Allocation: By constantly analyzing the state of warehouse operations, MARL can dynamically assign tasks to agents based on current needs and agent availability. This flexibility improves overall workflow efficiency and reduces bottlenecks.

Efficient Replenishment: Replenishment tasks are critical for ensuring that picking tasks can be carried out without delays. MARL optimizes the scheduling of replenishment to guarantee that high-demand items are always available at accessible locations.

Integration of Human and Robot Agents: One of the unique capabilities of MARL is its ability to integrate and coordinate between human and robotic agents. This hybrid approach leverages the strengths of both types of agents—robots for their speed and endurance, and humans for their problem-solving and decision-making skills.

The adaptation of multi-agent reinforcement learning in warehouse operations represents a significant advancement in the field of logistics. By optimizing task allocation through MARL, warehouses can achieve higher efficiency, better time management, and a seamless integration of human and robotic efforts. This technology adaptation not only enhances operational efficiency but also positions warehouses to meet the increasing demands of modern supply chains effectively.

3.12 A Technical Overview of MARL

Multi-Agent Reinforcement Learning (MARL) is a branch of artificial intelligence where multiple agents learn to interact with an environment to maximize their cumulative reward [10]. MARL is distinct from single-agent reinforcement learning because it involves coordination, cooperation, or competition among agents within the same environment. This complexity adds layers to the learning process, as each agent must consider not only the environment but also the potential actions and reactions of other agents.

Cooperative MARL: In cooperative MARL, all agents work together towards a common goal. The success of the system depends on the collective efforts of all agents. This approach is often used in scenarios like synchronized robot swarms or collaborative filtering systems.

Competitive MARL: Competitive MARL involves scenarios where agents compete against each other. Each agent aims to maximize its own rewards, possibly at the expense of other agents. Examples include strategic games like chess or poker.

Mixed MARL: This type involves elements of both cooperation and competition among agents. Agents might compete in sub-groups while cooperating within their groups. This is common in complex simulations involving multiple stakeholders with partially overlapping objectives.

3.12.1 Key Areas Under Multi-Agent Reinforcement Learning

Policy Gradient Methods: Policy gradient methods in MARL focus on learning a parameterized policy that maximizes the expected long-term reward through gradient ascent. These methods are advantageous in scenarios where the action space is continuous or when policies are explicitly stochastic. In MARL, policy gradient methods can be used to enable agents to learn cooperative strategies by optimizing shared objectives, making them particularly useful in environments where collaboration directly impacts the outcome, such as coordinated robot navigation.

Value-based Methods: Value-based approaches, like Q-learning, focus on learning the value of actions taken in given states. In MARL, value-based methods can be extended to multi-agent settings, such as Multi-Agent Q-Learning, where each agent learns to predict the value of its actions based on not only the state of the environment but also the anticipated actions of other agents. These methods are well-suited for environments where each agent's success is individually determined but is also affected by the actions of others.

Actor-Critic Methods: Actor-Critic methods in MARL combine the benefits of both policy gradient and value-based methods. Each agent has an actor that suggests actions based on the current policy and a critic that evaluates the action based on a value function. This approach allows agents to learn more stable policies by reducing the variance of policy updates. In MARL, Actor-Critic methods can be particularly effective for complex coordination tasks where both the evaluation of individual actions and the guidance toward optimal policies are crucial.

Model Predictive Control (MPC) Methods: MPC in MARL involves creating a model of the environment that predicts future states based on possible actions. Agents use this model to optimize their decision-making process by simulating different futures and choosing actions that lead to the

best predicted outcomes. This method is beneficial in structured environments with predictable dynamics and can be used for strategic planning in logistics and manufacturing processes where forecasting and long-term planning are vital.

A detailed Theoretical Overview related to Mult-Agent Systems and Reinforcement Learning can be found at Appendix A.

3.12.2 Use of Deep Q Networks in MARL for Task Assignment Problem

I have chosen to apply Deep Q-Networks (DQN), a specific area within MARL, to solve the problem of task allocation in modern warehouses that use both humans and robots. DQN integrates deep learning with Q-learning, where an artificial neural network approximates the Q-value function. This method is particularly effective in environments with high-dimensional state spaces, such as warehouses.

As for reasons of choosing DQN for implementing a solution following can be stated:

Handling High-Dimensional Data: Warehouses involve complex dynamics with numerous variables including inventory levels, task priorities, agent and device availability. DQN is well-suited for such environments because it can process and make decisions based on large volumes of data.

Ability to Learn from Experience: DQN allows agents to learn optimal policies directly from raw sensory data gathered through interactions with the environment, making it ideal for dynamic and unpredictable warehouse settings.

Bridging Research Gaps: Current research often focuses on either purely cooperative or competitive MARL environments. However, warehouses require a nuanced approach where agents cooperate (humans and robots working towards common logistical goals) but also compete (for resources like space and time). DQN provides a flexible framework that can be adapted for these mixed cooperative-competitive scenarios, addressing a significant gap in existing research.

3.13 Summary

Throughout the exploration of various technologies in this section, I have reviewed multiple approaches that have been applied to warehouse operations and task assignments. These included several branches of multi-agent reinforcement learning (MARL), each offering distinct advantages and limitations depending on the specific requirements of the warehouse environment.

Deep Q-Networks (DQN), a sophisticated area of MARL, has been identified as the most suitable technology for addressing the challenges associated with the dynamic and complex environment of modern warehouses that employ both human and robotic workers. DQN stands out due to its ability to handle high-dimensional state spaces and to learn optimal policies from large volumes of operational data. This capability makes it particularly effective for environments where rapid decision-making and adaptability are critical.

The choice of DQN within the framework of MARL is justified by its proven efficiency in environments that require both cooperative and competitive interactions among multiple agents.

By integrating DQN, I aim to bridge the existing research gap in real-time adaptive task allocation in warehouses, enhancing operational efficiency and responsiveness to fluctuating demands and priorities.

The next section will delve into the "Approach" used to implement DQN with MARL in the warehouse setting, detailing the methodology, experimental setup, and anticipated challenges in integrating this technology into existing warehouse operations. This approach is expected to significantly improve task allocation efficiency, thereby optimizing the throughput and reducing operational delays in modern warehouse environments.

CHAPTER 4

AN APPROACH TO WAREHOUSE TASK ASSIGNMENT USING MARL

4.1 Introduction

Following the theoretical foundation laid in the previous chapters on reinforcement learning and multi-agent systems, this section proposes a MARL-based approach to optimize task assignments in modern warehouses employing both human workers and robots. The focus here is strictly on the allocation of tasks—particularly picking and replenishment tasks—which are critical for efficient warehouse operations. This approach leverages the capabilities of MARL to manage task allocation dynamically, aiming to enhance overall warehouse efficiency.

4.2 Hypothesis

I hypothesize that DQN within Multi-Agent Reinforcement Learning (MARL) can be used to implement a solution for warehouse task assignment which employees a mix workforce of humans and robots with varying capabilities.

4.3 Inputs

The primary inputs for the MARL system are detailed states of the warehouse environment, comprising two main categories: current available tasks and available devices.

Tasks: Each task in the warehouse is defined by multiple attributes: Task_ID, Type (pick, replenish, load, put), product details, quantity, locations (from and to), anticipated completion time, order ID (if applicable), and current status (available, active, or done).

Devices: Each device available for task completion is characterized by Device_ID, type (Pallet Jacks for picking, Forklifts for other tasks), status, and the Task_ID of the current assigned task.

These inputs are continuously fed into the MARL system, allowing each agent to assess the current state and make informed decisions on task selection based on the availability of tasks and the necessary devices.

4.4 Output

The output of the MARL system is an optimized task allocation that prioritizes efficiency and task completion, particularly focusing on outbound tasks such as picking. The effectiveness of the system is evaluated through several key performance indicators (KPIs):

1. Pick Task Completion Efficiency: Measures the timeliness of completing pick tasks.
2. Task Distribution Among Agents: Ensures a balanced workload among agents, promoting fairness and efficiency.

3. Average Number of Idle Agents: Lower numbers indicate higher efficiency in task allocation.
4. Average Number of Idle Devices: Similarly, lower numbers suggest better utilization of available resources.

4.5 Process

The core of the approach is the decision-making process undertaken by agents, guided by a deep reinforcement learning algorithm (DQN). Agents are trained to select tasks that optimize a predefined reward mechanism.

Rewards: Completing a pick task earns 3 points, a replenishment task required for a pick task earns 2 points, and other tasks earn 1 point. Selecting a task without available devices, or choosing a task that is already active or completed, results in a penalty of -1 points.

This reward system compels agents to make strategic decisions that align with operational priorities, such as completing pick tasks swiftly and ensuring necessary replenishments are done in time to facilitate these picks.

KPI calculation: The simulation environment developed using the OpenAI Gym toolkit not only facilitate the reinforcement learning process but also to track and record necessary data for KPI evaluation. As agents execute their tasks, every action and its outcome are logged, allowing the system to calculate key performance indicators that measure the efficiency and effectiveness of task distribution and completion. At the conclusion of each simulation run, the environment computes the KPI values such as Pick Task Completion Efficiency, Task Distribution Among Agents, Average Number of Idle Agents, and Average Number of Idle Devices. These metrics are then presented as outputs, providing administrators with valuable insights into the operational performance and highlighting areas for potential improvement. This integrated tracking and reporting capability ensures that the simulation environment is not just a training ground for AI agents but also a powerful analytical tool for strategic decision-making in warehouse management.

4.6 Users

The system is primarily designed for use by warehouse administrators and workers, enhancing their ability to manage and execute warehouse tasks effectively. Administrators can monitor performance through the KPIs, while workers receive real-time task assignments optimized for current conditions, reducing idle times and improving task distribution.

4.6 Features

The developed MARL system includes several innovative features:

- **Simulation Environment**: Developed using the OpenAI Gym toolkit, this environment allows for extensive testing and refinement of the MARL algorithms in a controlled setting.

- **Dynamic Task Allocation**: The system dynamically assigns tasks based on current warehouse states, aiming to maximize efficiency and reduce bottlenecks.
- **Adaptive Learning**: Agents continually learn and adapt their strategies based on ongoing feedback from the environment, improving their decision-making over time.
- **User-Friendly Interface**: Administrators have access to a dashboard that displays real-time data on task status, agent activity, and overall warehouse efficiency.

4.8 Summary

This approach sets the stage for a sophisticated application of MARL in warehouse management, specifically focusing on task assignment without the complexities of navigation or collision avoidance. The next chapter will delve into the design and implementation details of the proposed approach, highlighting how theoretical concepts from earlier chapters are applied to achieve practical outcomes in a real-world setting.

CHAPTER 5

MARL BASED DESIGN FOR TASK ALLOCATION IN HYBRID WAREHOUSES

5.1 Introduction

In the exploration of the Multi-Agent Reinforcement Learning (MARL) approach to warehouse task allocation, I emphasize the design of a system that prioritizes efficiency and task execution speed. This chapter details the architecture that underpins the solution, offering a modular perspective and highlighting the synergy between human and robotic agents in a technologically advanced warehouse setting.

5.2 Top-level architecture

The system's architecture, encapsulated in the "MARL-based Warehouse Task Assignment Framework," presents a holistic view of the components and their interactions. The framework consists of several modules, each serving a distinct purpose in facilitating the coordination of tasks among agents. The core of the design lies in the adaptiveness and responsiveness of these modules, working collectively to optimize task allocation.

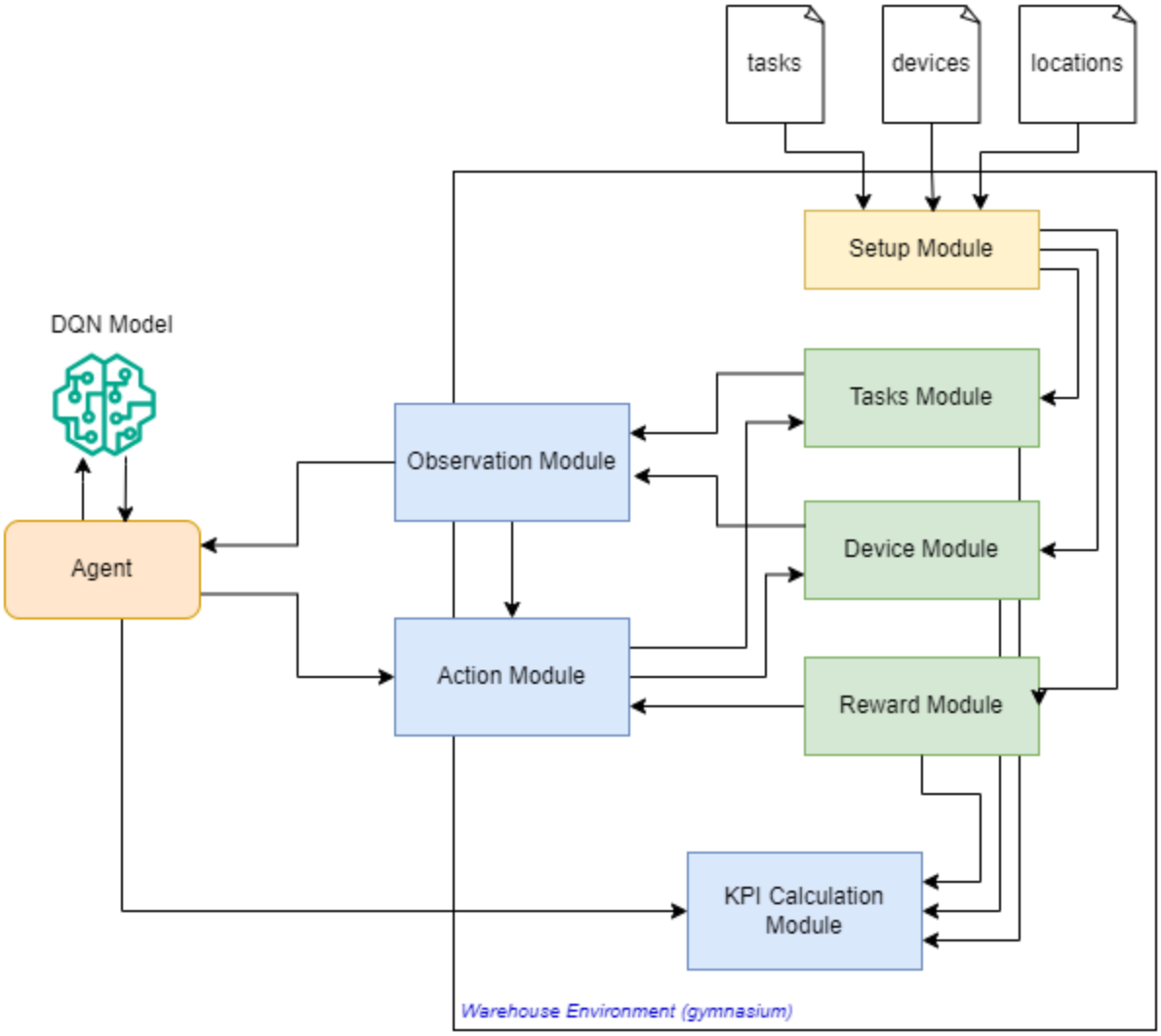

Figure 5.1 – MARL based design for warehouse task assignment

Each module within the design plays a specific role in contributing to the overall effectiveness of the task allocation system. The following descriptions focus on the functionality of the modules rather than the technical intricacies of their implementation.

5.3 Setup Module

The Setup Module is the initial point of engagement in the system, orchestrating the environment by integrating task specifications, device availability, and the layout of the warehouse from structured data sources like CSV files. This module distinguishes between task types—prioritizing pick and replenishment tasks—and ensures the smooth onboarding of resources. It plays a pivotal

role in defining parameters that mirror the operational conditions of a real-world warehouse, thus allowing for an authentic simulation experience.

5.4 Tasks Module

The Tasks Module operates as the control center for all task-related operations. It holds the responsibility of maintaining an up-to-date ledger of tasks, including their status, priority, and which agent is assigned to each task. This module not only keeps track of pending tasks but also archives completed tasks to inform future decision-making processes. Its dynamic updating mechanism ensures that the system adapts to changes in task urgency and availability, thereby maintaining a streamlined workflow.

5.5 Device Module

The Device Module is tasked with the inventory and allocation of devices crucial to task execution. It maintains an accurate record of the location, availability, and engagement of Pallet Jacks and Forklifts. Human agents consult this module to secure the necessary equipment for their assigned tasks, while the module itself ensures that equipment utilization is optimized to reduce idle times and enhance overall operational efficiency.

5.6 Reward Module

Functioning as the feedback generator, the Reward Module assesses the actions taken by agents and assigns rewards based on their effectiveness. This module underpins the learning algorithm by differentiating between high and low-value actions, thereby directing agents towards behaviors that contribute to faster and more accurate task completion. The sophisticated reward system designed in this module aligns with the key objectives of warehouse operations, such as task prioritization and timely execution.

5.7 KPI Calculation Module

The KPI Calculation Module is the analytical brain of the system, dedicated to evaluating performance through various KPIs. It meticulously measures factors such as task completion rate, device usage efficiency, and overall time to complete orders. By quantifying these metrics, the module provides objective insights into the system's effectiveness, enabling stakeholders to pinpoint areas of success and opportunities for improvement.

5.8 Action Module

The Action Module serves as the interface through which agents execute tasks. It interprets the policy decisions made by the DQN and translates them into actionable steps within the simulation environment. This module ensures that each action taken by an agent is reflective of the strategy

formulated through the reinforcement learning process, aiming to enhance the precision of task allocation and execution.

5.9 Observation Module

Critical for situational awareness, the Observation Module offers a real-time snapshot of the warehouse environment. It collects and distributes information regarding the status of tasks, the position and engagement of agents, and the availability of devices. Agents rely on the data from this module to make informed decisions, and it is integral for the DQN to assess the current state of the system when formulating its next move.

5.10 Agent

Agents are the dynamic entities within the system, designated to carry out the tasks. Human workers and robots are modeled distinctly, each with their tailored DQN, ensuring their roles in task assignments are executed considering their respective capabilities. This delineation allows for the nuanced training of agents based on the specificities of their functions within the warehouse.

5.11 DQN (Deep Q Network)

At the core of each agent is the Deep Q Network (DQN), the decision-maker that learns and iterates over time. By processing the rewards and observations fed into it, the DQN continually refines its policy, striving for optimal task allocation and completion. This evolving intelligence is what drives the system towards enhanced efficiency and is indicative of the potential for artificial intelligence to transform operational strategies in real-world settings.

5.8 Summary

The design presented here lays the groundwork for an intelligent and responsive warehouse task allocation system. Key modules such as the Task, Device, and Reward modules work in concert to ensure that the most critical tasks are prioritized, and resources are allocated optimally. This design sets the stage for the next chapter, where I delve into the implementation details, showcasing how the design principles are brought to life through coding, algorithms, and real-world application. I invite the reader to proceed with an anticipation of how these designs manifest in a functioning system, optimizing warehouse operations and task assignments.

CHAPTER 6

IMPLEMENTATION OF MARL BASED WAREHOUSE TASK ASSIGNMENT

6.1 Introduction

Following the presentation of the proposed solution's design in the previous chapter, this chapter delves into the practical aspects of its implementation. Emphasizing the relationship between design and realization, the implementation encapsulates the translation of theoretical modules into functioning components, utilizing appropriate software, hardware, and algorithmic strategies. This chapter will navigate through the construction of each module depicted in the design diagram, ensuring a clear connection between design intention and practical execution.

6.2 Setup Module Implementation

The Setup Module forms the foundation of our warehouse simulation environment. It begins by initializing the parameters necessary for training or simulation. This entails reading and processing information from CSV files, including task descriptions, device statuses, and location coordinates. The programming language of choice, due to its extensive library support for data manipulation and its compatibility with machine learning frameworks, is Python. Using the “pandas” library, the module ingests data, converts it into an internal format suitable for real-time access during the simulation, and establishes initial states.

Tools and Technologies:

- Programming language: Python
- Libraries: Pandas for data handling
- CSV files as data input

6.3 Tasks and Device Module Implementation

Tasks Module and Device Module are the central management systems, handling the dynamic aspects of warehouse operations. They ensure the continual updating of task statuses and device availabilities, responding to the demands of the environment. For these modules, Python scripts equipped with data structures, such as dictionaries and queues, enable efficient tracking and updating. These structures are manipulated with thread-safe operations to handle concurrent access by multiple agents.

Status Encodings

- At the beginning the status of a task is “available” and this is represented (encoded) with number “1”. When a task is active, the value will change into 2. When a task is completed, the encoded value is “-1”.
- At the beginning the status of all the devices is “available” and this is represented (encoded) with number “1”. When a device is active, the value will change into 2 and when it is free again (the agent has completed the task using that device) it will change back to 1.

6.5 Reward Module Implementation

The Reward Module is responsible for quantifying the success of agents' actions. The implementation is built using a reward function in Python that maps the outcome of actions to numerical rewards, thus providing feedback to the agents. This function is designed to reinforce desired behaviors, like task completion, and penalize unproductive actions, ensuring agents learn to optimize task assignment effectively.

Algorithm:

- Plus 3 for pick task completion
- Plus 2 for required replenishment tasks
- Plus 1 for other tasks
- Minus 1 for selecting tasks without available devices or already active/completed tasks

6.6 KPI Module Implementation

The KPI Calculation Module is the analytic backbone, assessing the operational efficiency through various Key Performance Indicators (KPIs). This module leverages statistical functions in Python to compute metrics such as task completion rates and device utilization, which serve as a measure of the simulation's success.

KPIs:

- Pick task completion efficiency.
- Task distribution among agents.
- Average free agents.
- Average free devices.

Pick Task Completion Efficiency Pseudo Code:

```
function calculatePickTaskCompletionEfficiency(tasks):
    lastPickTaskTime = 0
    for task in tasks:
        if task.type == "pick" and task.status == "done":
            lastPickTaskTime = max(lastPickTaskTime, task.completionTime)
    totalTaskTime = currentTime - startTime
    efficiency = lastPickTaskTime / totalTaskTime
```

```
return efficiency
```

Task Distribution among Agents Pseudo Code:

```
function calculateTaskDistribution(agents):
    totalTasks = sum(agent.completedTasks for agent in agents)
    averageTasksPerAgent = totalTasks / len(agents)
    distribution = [abs(agent.completedTasks - averageTasksPerAgent) for agent in agents]
    distributionScore = 1 - (sum(distribution) / totalTasks)
    return distributionScore
```

So here when the distributionScore is low, it is bad (not distributed evenly)

Average Free Agents Pseudo Code:

```
function calculateAverageFreeAgents(agentStatusLogs, totalSimulationTime):
    totalFreeAgentTime = sum(log.freeTime for log in agentStatusLogs)
    averageFreeAgents = totalFreeAgentTime / totalSimulationTime
    return averageFreeAgents
```

Average Free Devices Pseudo Code:

```
function calculateAverageFreeDevices(deviceStatusLogs, totalSimulationTime):
    totalFreeDeviceTime = sum(log.freeTime for log in deviceStatusLogs)
    averageFreeDevices = totalFreeDeviceTime / totalSimulationTime
    return averageFreeDevices
```

6.7 Action and Observation Module Implementation

The Action Module provides agents with the interface to perform actions within the environment, while the Observation Module offers a real-time snapshot of the warehouse's current state. The implementation utilizes a combination of Python's object-oriented capabilities and its multiprocessing library to facilitate asynchronous interactions between agents and the simulation environment.

Tools and Technologies:

- Integration with the DQN Model
- Methods for task selection and completion

6.9 Agent and DQN Implementation

Agents, both human and robotic, are equipped with a Deep Q-Network (DQN), a reinforcement learning algorithm realized through a neural network implemented with the machine learning framework TensorFlow (Keras). The DQN model enables agents to learn and refine their task assignment strategies over time. The neural network's architecture is crafted to process the state inputs (tasks and devices) and output the optimal action for task selection. Training of these models is done through simulation episodes, with the reward system guiding the learning process.

Components:

- DQN Model
- Learning algorithm
- Decision-making process

Model Information:

- 2 input layers (for tasks and devices)
- 2 hidden layers with 120 and 60 neurons in each
- One output layer with the length of number of tasks
- Activation: ReLu (Rectified Linear Units)
- Optimizer: Adam

You can find out the model Implementation code in the Appendix B.

6.10 Summary

The implementation of a Multi-Agent Reinforcement Learning-based warehouse task assignment system is a meticulous process that transforms theoretical constructs into a practical, functioning system. Through careful development of each module, the implementation is designed to optimize task allocation, improve efficiency, and ensure smooth operations within a modern warehouse environment.

This chapter sets the stage for the forthcoming evaluation, where the real-world applicability and effectiveness of the developed system will be rigorously assessed. The reader is encouraged to proceed to the evaluation chapter for insights into the performance and benefits of the implementation in a practical context.

CHAPTER 7

EVALUATION

7.1 Introduction

This chapter describes the evaluation strategy used to assess the performance of the multi-agent reinforcement learning (MARL) approach for warehouse task assignment. The evaluation was conducted through the implemented simulation environment that replicates a warehouse setting with human workers and robots. The warehouse environment was designed to include tasks, devices (forklifts and pallet jacks), and agents (human workers and robots). The agents were trained using a Deep Q-Network (DQN) to make decisions on task selection, while considering factors like task type, device availability, and reward incentives.

7.2 Evaluation Strategy

The evaluation aimed to investigate the effectiveness of the MARL approach in improving warehouse efficiency by:

- Prioritizing pick tasks to meet order fulfillment deadlines.
- Balancing task distribution among agents to avoid overburdening specific workers or robots.
- Minimizing idle time for both agents and devices.

These will be evaluated with the following KPIs that the environment outputs after each simulation run.

1. Pick Task Completion Efficiency: This KPI measures the effectiveness of the system in prioritizing and completing pick tasks, which are crucial for meeting order fulfillment deadlines.

2. Task Distribution Among Agents: This KPI assesses how evenly tasks are distributed among all agents (human workers and robots) in the warehouse. A balanced distribution prevents overburdening specific agents and ensures everyone contributes fairly.

3. Average Free Agents in the Whole Time: This KPI measures the average number of agents who are available for task assignment throughout the simulation. A lower value indicates that most agents are actively working on tasks, minimizing idle time.

4. Average Free Devices in the Whole Time: This KPI similarly measures the average number of devices (forklifts and pallet jacks) that are not currently in use during the simulation. A lower value indicates that devices are being utilized efficiently to support task completion.

7.3 Experimental Setup

The evaluation strategy consisted of three main simulation settings:

a. **Simulation Setting 1**: This setting involved only human workers to evaluate the effectiveness of MARL in a human-only warehouse environment.
b. **Simulation Setting 2**: This setting incorporated both human workers and robots to assess the efficacy of MARL in a mixed-agent warehouse environment.
c. **Simulation Setting 3**: This setting incorporated both human workers and robots, but had only very low devices numbers to work on. This setting would explore how the agents handle situations where device availability becomes a factor in task selection.

Simulation Setting 1 Configuration:

- Tasks: 50 (20 – pick, 10 – replenishment, 10 – put, 10 – load)
- Humans: 4, Robots: 0
- Devices: 6 (3 pallet jacks, 3 forklifts)

Simulation Setting 2 Configuration:

- Tasks: 50 (20 – pick, 10 – replenishment, 10 – put, 10 – load)
- Humans: 4, Robots: 4
- Devices: 6 (3 pallet jacks, 3 forklifts)

Simulation Setting 3 Configuration:

- Tasks: 50 (20 – pick, 10 – replenishment, 10 – put, 10 – load)
- Humans: 4, Robots: 4
- Devices: 3 (3 forklifts)

Within each of the above-described simulation setting, three scenarios were implemented:

d. **Scenario 1**: Random Policy: In this scenario, agents randomly selected tasks from the available pool. This scenario served as a baseline to compare the performance of the MARL approach.
e. **Scenario 2**: Partial DQN Adoption: In this scenario, only a subset of agents used the DQN-based policy for task selection, while others continued with random selection.
f. **Scenario 3**: Full DQN Adoption: In this scenario, all agents employed the DQN policy for task assignment.

7.4 Results

7.4.1 Results of Simulation 1

Setting: Tasks: 50 (20 – pick, 10 – replenishment, 10 – put, 10 – load), Humans: 4, Robots: 0, Devices: 6 (3 pallet jacks, 3 forklifts)

Scenario 1.1 (Random Policy):

KPI 1: Pick Task Completion Efficiency = (1 - 263/284) = 0.074

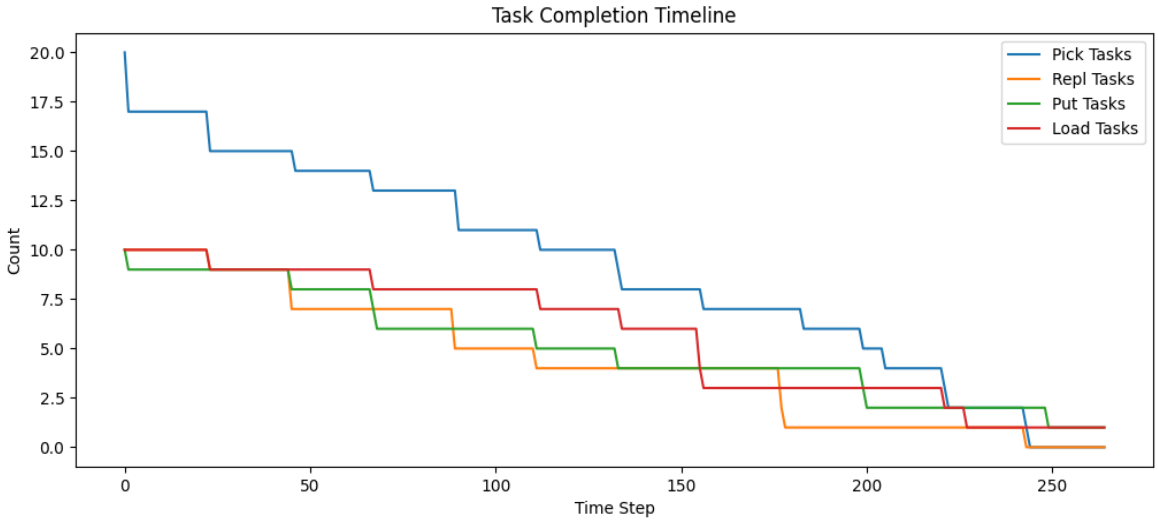

Figure 7.1: Setting 1 – Scenario 1 Task Completion Chart

- KPI 2: Task Distribution Among Agents = 0.2
- KPI 3: Average free agents = 0.215
- KPI 4: Average free devices = 2.215

Scenario 1.2 (Half of Agents using DQN):

KPI 1: Pick Task Completion Efficiency = (1 - 213/284) = 0.25

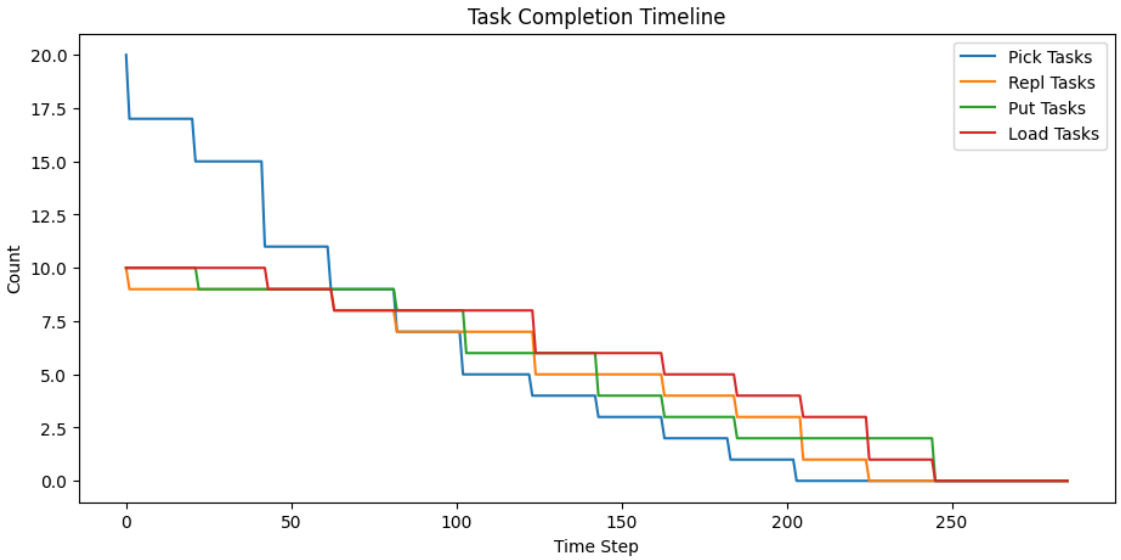

Figure 7.2: Setting 1 – Scenario 2 Task Completion Chart

- KPI 2: Task Distribution Among Agents = 0.1
- KPI 3: Average free agents = 0.195
- KPI 4: Average free devices = 2.628

Scenario 1.3 (All Agents using DQN):

KPI 1: Pick Task Completion Efficiency = (1 - 180/284) = 0.366

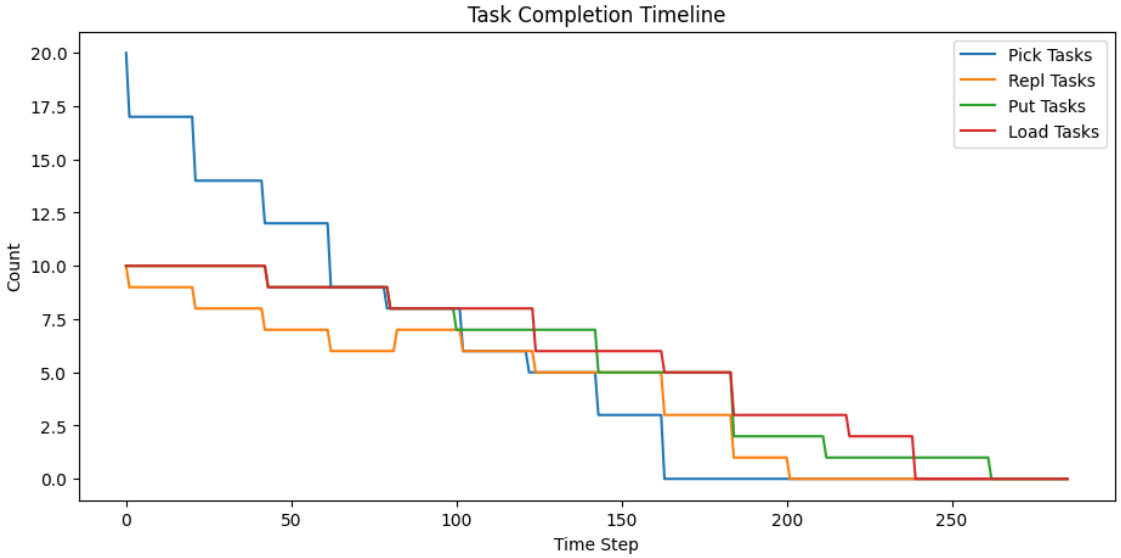

Figure 7.3: Setting 1 – Scenario 3 Task Completion Chart

- KPI 2: Task Distribution Among Agents = 0.15
- KPI 3: Average free agents = 0.15
- KPI 4: Average free devices = 2.052

7.4.2 Results of Simulation 2

Setting: 50 (20 – pick, 10 – replenishment, 10 – put, 10 – load), Humans: 4, Robots: 4, Devices: 6 (3 pallet jacks, 3 forklifts)

Scenario 2.1 (Random Policy):

KPI 1: Pick Task Completion Efficiency = (1 - 120 / 239) = 0.285

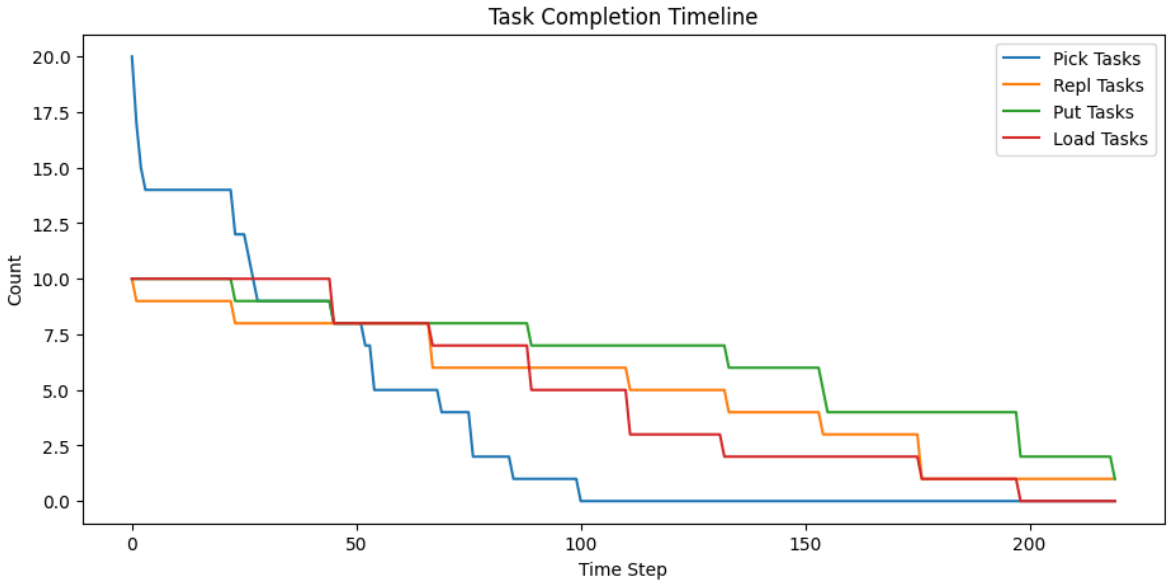

Figure 7.4: Setting 2 – Scenario 1 Task Completion Chart

- KPI 2: Task Distribution Among Agents = 0.4
- KPI 3: Average free agents = 2.509
- KPI 4: Average free devices = 2.75

Scenario 2.2 (Half of Agents using DQN):

KPI 1: Pick Task Completion Efficiency = (1 - 98 / 239) = 0.589

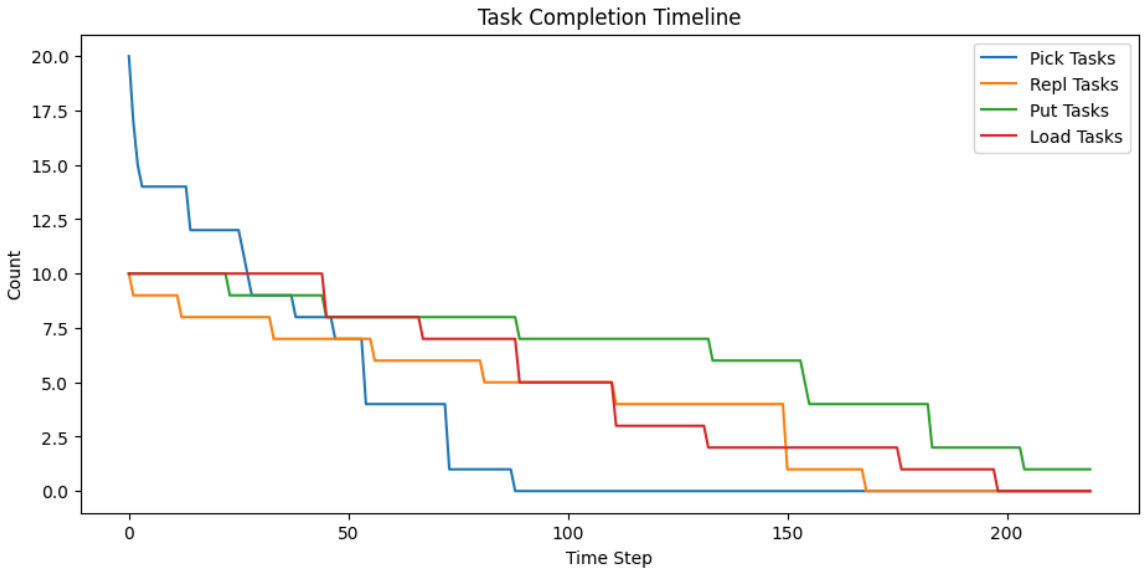

Figure 7.5: Setting 2 – Scenario 2 Task Completion Chart

- KPI 2: Task Distribution Among Agents = 0.21
- KPI 3: Average free agents = 2.602
- KPI 4: Average free devices = 3.26

Scenario 2.3 (All Agents using DQN):

KPI 1: Pick Task Completion Efficiency = (1 - 88 / 239) = 0.631

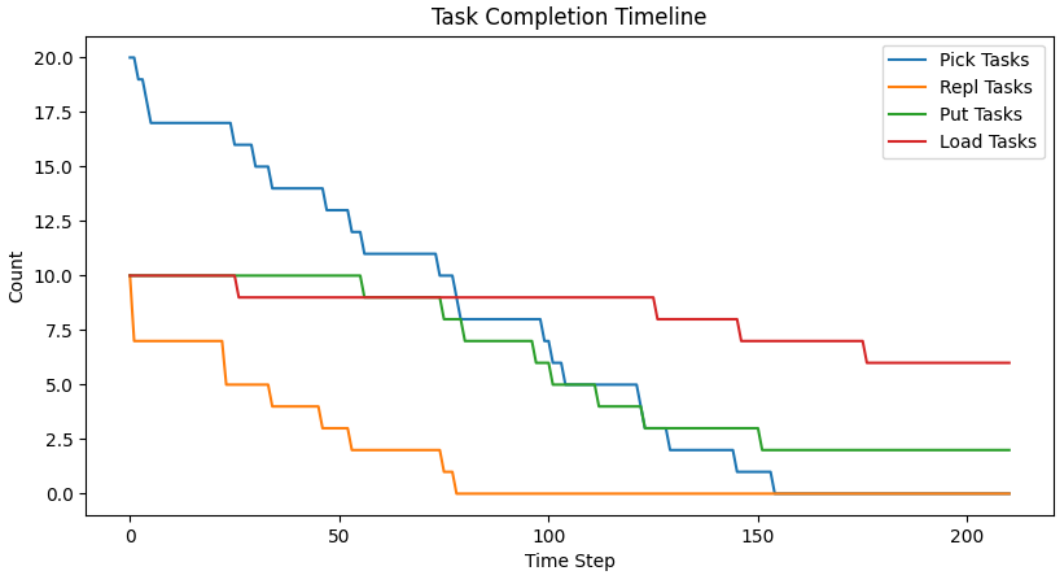

Figure 7.6: Setting 2 – Scenario 3 Task Completion Chart

KPI 2: Task Distribution Among Agents = 0.15

KPI 3: Average free agents = 2.91

KPI 4: Average free devices = 3.86

7.4.3 Results of Simulation 3

Setting: 50 (20 – pick, 10 – replenishment, 10 – put, 10 – load), Humans: 4, Robots: 4, Devices: 3 (3 forklifts).

Scenario 3.1 (Random Policy):

KPI 1: Pick Task Completion Efficiency = (1 - 168/178) = 0.056

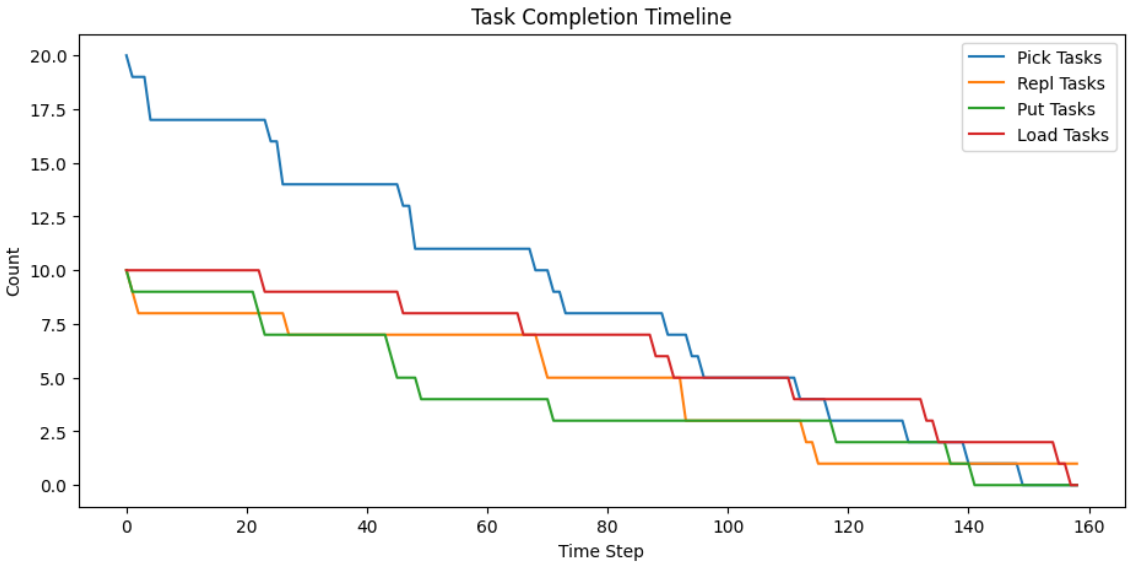

Figure 7.7: Setting 3 – Scenario 1 Task Completion Chart

KPI 2: Task Distribution Among Agents = 0.6

KPI 3: Average free agents = 1.55

KPI 4: Average free devices = 0.842

Scenario 3.2 (Half of Agents using DQN):

KPI 1: Pick Task Completion Efficiency = (1 - 160/178) = 0.101

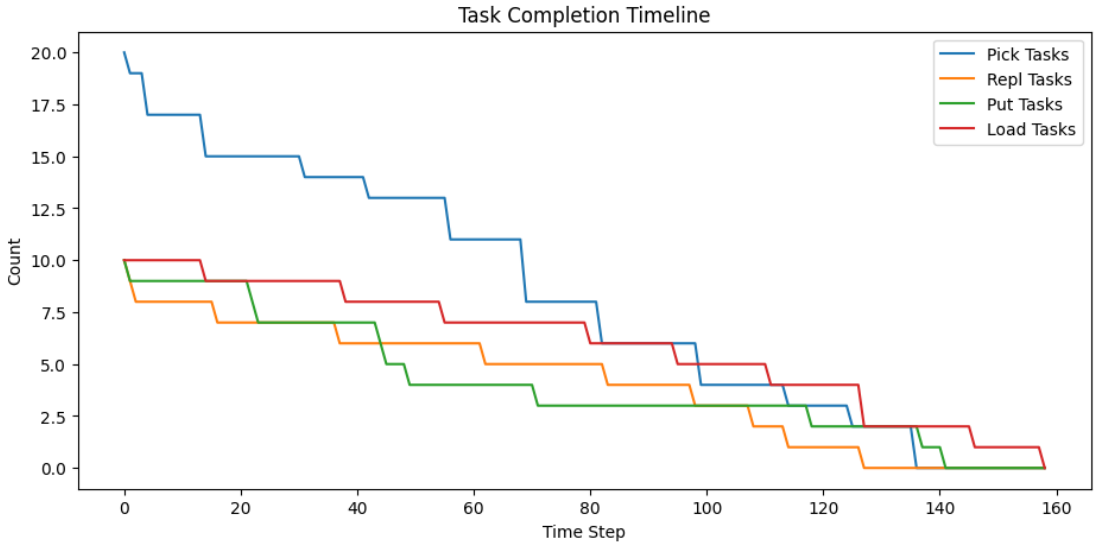

Figure 7.8: Setting 3 – Scenario 2 Task Completion Chart

KPI 2: Task Distribution Among Agents = 0.4

KPI 3: Average free agents = 1.43

KPI 4: Average free devices = 0.813

Scenario 3.3 (All Agents using DQN):

KPI 1: Pick Task Completion Efficiency = (1 - 128/178) = 0.280

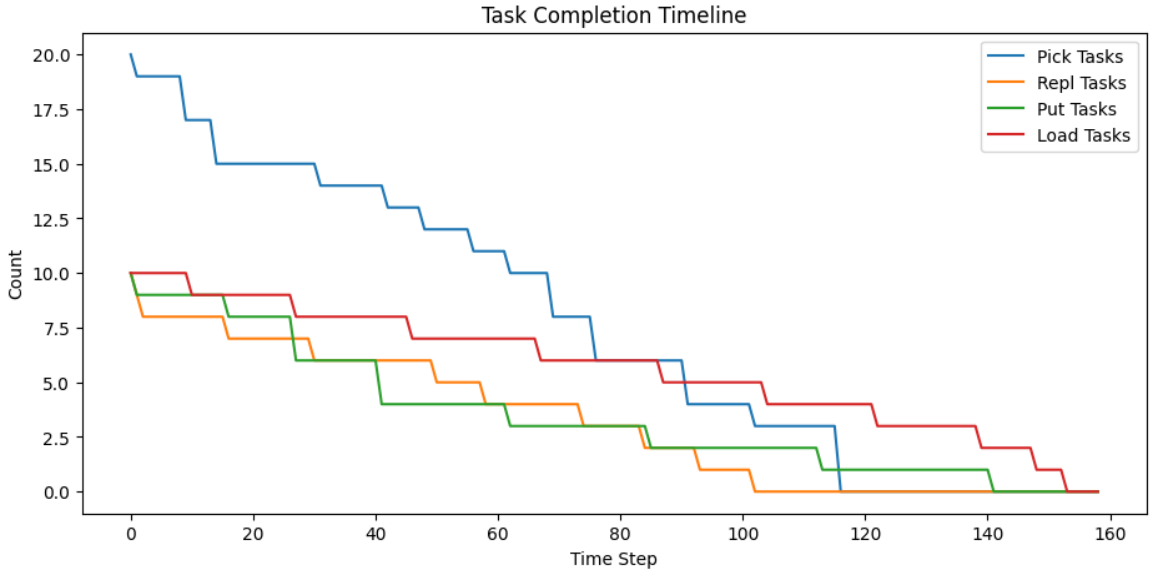

Figure 7.9: Setting 3 – Scenario 3 Task Completion Chart

KPI 2: Task Distribution Among Agents = 0.24

KPI 3: Average free agents = 1.63

KPI 4: Average free devices = 0.95

7.5 Developed Simulation Environment

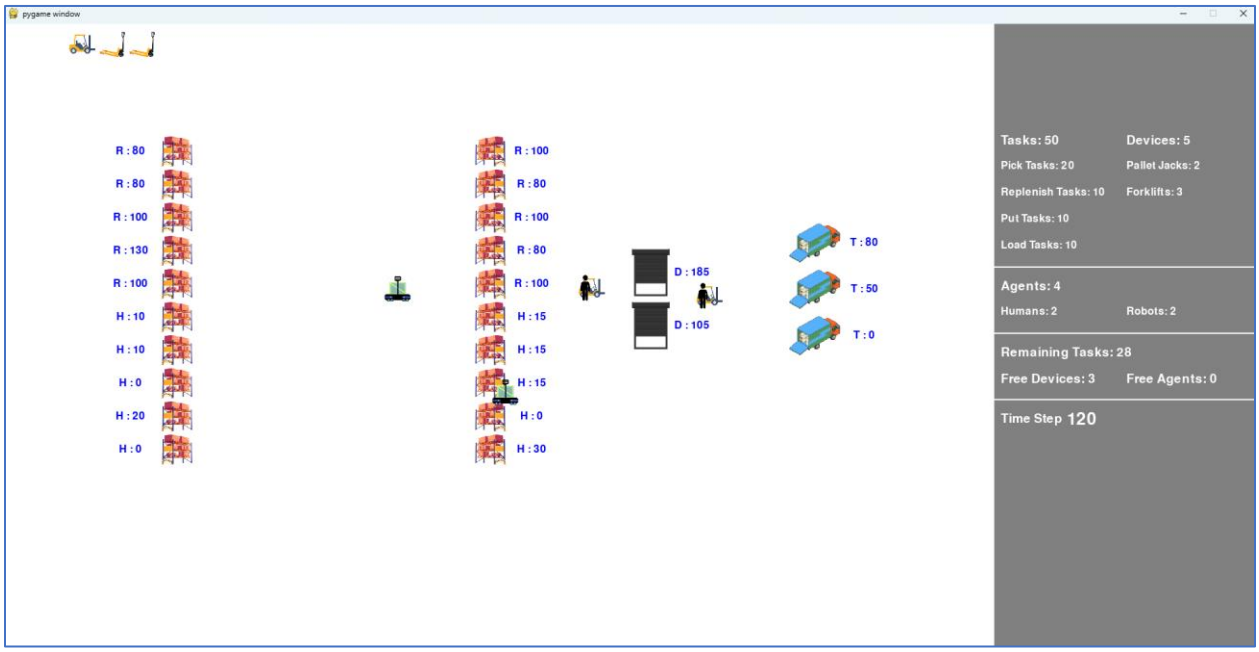

Figure 7.10: Developed Virtual Environment Using Gymnasium and pygame Libraries

CHAPTER 8

CONCLUSION AND FURTHER WORK

8.1 Introduction

This research explored how multi-agent reinforcement learning can be used to assign tasks in a warehouse with both human and robot workers. The goal was to improve the efficiency of warehouse operations by prioritizing pick tasks. A multi-agent DQN system was developed to select tasks for agents. The system was evaluated in a simulated warehouse environment. The results showed that using DQN agents improved pick task completion efficiency compared to using random agents.

8.2 Overall conclusion

Reflecting on the evaluation results, our MARL-based system has demonstrated a substantial improvement in warehouse task assignment, particularly in prioritizing and completing pick tasks more efficiently. I estimate the system's overall effectiveness at a commendable 27% improvement averagely in all simulations settings, a significant leap forward in optimizing warehouse pick task operations.

Pick Tasks Completion Efficiency				
	Zero DQN Agents	Half DQN Agents	All are DQN Agents	Improvement %
Setting 1	0.074	0.25	0.306	23.2
Setting 2	0.285	0.589	0.631	34.6
Setting 3	0.056	0.101	0.28	22.4

Figure 8.1: Pick Task Completion Efficiency Summary

8.3 objective-wise achievements

The systematic exploration of the project yielded the following accomplishments aligned with the predefined objectives:

- A critical examination of existing warehouse processes and task assignment strategies highlighted the need for innovation in this domain. My research identified gaps in task prioritization and efficiency, emphasizing the potential of a MARL-based system to improve these aspects.

- The review of technologies used for warehouse task assignment strategies provided a comprehensive understanding of current methodologies. It became evident that traditional strategies lacked the adaptability and real-time decision-making capabilities that MARL systems can offer.
- The innovation of a MARL-based system aimed at efficient task distribution and device utilization has been successfully realized. The system has proven its capability in prioritizing urgent tasks and optimizing the use of available devices, as evidenced by the improved task completion efficiency in various simulation settings.
- The evaluation of the system's performance through Key Performance Indicators (KPIs) demonstrated significant improvements. The data reflects an increase in pick task completion efficiency across all settings when MARL-trained agents are utilized, with improvements ranging from 22.4% to 34.6% over scenarios employing zero DQN agents.

8.4 Limitation

Throughout the development of the MARL-based warehouse task assignment system, I faced several significant challenges and identified key limitations that influenced our research journey.

One of the primary challenges was the considerable amount of time required to develop and prepare the warehouse simulation environment. Crafting a simulation that accurately represents the intricate workings of a real-world warehouse was a complex and demanding task. It involved a deep dive into the operational details and required meticulous attention to the creation of a virtual environment that could both represent the physical warehouse and support the learning algorithms effectively.

Moreover, devising a practical and effective reward system presented its own set of complexities. The initial phases of establishing a reward mechanism that would not only encourage efficient task completion but also balance device utilization were particularly challenging. It required a thorough understanding of warehouse operations and the intricacies of task interdependencies. Developing a system that could dynamically adapt to the operational states of the warehouse, and differentiate between the priority of tasks, took substantial effort and iterations to refine.

These challenges, while significant, contributed to the depth of my research and the robustness of the final system. They highlight areas where future projects may benefit from a more streamlined approach or the application of different strategies to reduce the initial development time and enhance the reward mechanism's effectiveness from the outset.

8.5 Further work

Looking ahead, several avenues for enhancement and exploration present themselves:

- Advance the DQN to encompass decision-making for both task and device selection, creating a more autonomous and holistic approach to warehouse task assignment.

- Integrate warehouse layout considerations into the DQN decision-making process, which could further optimize task assignments in relation to spatial efficiency.
- Expand the simulation settings to include varying task times, potential device breakdowns, and human break schedules, adding layers of realism to the model.
- Investigate the effects of scaling the number of tasks and devices within the simulation to understand the model's limits and scalability.

8.6 Summary

In summary, while this research has successfully implemented a MARL-based approach for task allocation in a warehouse setting, yielding significant improvements in efficiency, the system's current iteration leaves room for future work. An enhanced DQN model that encompasses device selection and accounts for the physical layout of the warehouse could pave the way for an even more sophisticated and efficient allocation system. This research lays the groundwork for such advancements, offering a solid foundation for subsequent innovation in the realm of warehouse logistics.

REFERENCES

1. Tubis, Agnieszka A., and Juni Rohman. "Intelligent Warehouse in Industry 4.0—Systematic Literature Review." *Sensors* 23, no. 8 (January 2023): 4105. https://doi.org/10.3390/s23084105.
2. Ganbold, Odkhishig, Kaustav Kundu, Haobin Li, and Wei Zhang. "A Simulation-Based Optimization Method for Warehouse Worker Assignment." *Algorithms* 13, no. 12 (December 2020): 326. https://doi.org/10.3390/a13120326.
3. Geng, Sai, Lei Wang, Dongdong Li, Benchi Jiang, and Xueman Su. "Research on Scheduling Strategy for Automated Storage and Retrieval System." *CAAI Transactions on Intelligence Technology* 7, no. 3 (2022): 522–36. https://doi.org/10.1049/cit2.12066.
4. Ács, Botond, László Dóra, Olivér Jakab, Alpár Jüttner, Péter Madarasi, and László Z. Varga. "Optimizations of a Multi-Agent System for a Real-World Warehouse Problem." *SN Computer Science* 3, no. 6 (August 8, 2022): 431. https://doi.org/10.1007/s42979-022-01296-6.
5. Burggräf, Peter, Johannes Wagner, Till Saßmannshausen, Dennis Ohrndorf, and Karthik Subramani. "Multi-Agent-Based Deep Reinforcement Learning for Dynamic Flexible Job Shop Scheduling." *Procedia CIRP* 112 (2022): 57–62. https://doi.org/10.1016/j.procir.2022.09.024.
6. Jiang, Chengzhi, and Zhaohan Sheng. "Case-Based Reinforcement Learning for Dynamic Inventory Control in a Multi-Agent Supply-Chain System." *Expert Systems with Applications* 36, no. 3 (April 2009): 6520–26. https://doi.org/10.1016/j.eswa.2008.07.036.
7. Krnjaic, Aleksandar, Raul D. Steleac, Jonathan D. Thomas, Georgios Papoudakis, Lukas Schäfer, Andrew Wing Keung To, Kuan-Ho Lao, et al. "Scalable Multi-Agent Reinforcement Learning for Warehouse Logistics with Robotic and Human Co-Workers." arXiv, July 7, 2023. http://arxiv.org/abs/2212.11498.
8. Schäfer, Lukas. "Task Generalisation in Multi-Agent Reinforcement Learning," 2022.
9. Khirwar, Madhav, Karthik S. Gurumoorthy, Ankit Ajit Jain, and Shantala Manchenahally. "Cooperative Multi-Agent Reinforcement Learning for Inventory Management," 2023. https://doi.org/10.48550/ARXIV.2304.08769.
10. Yang, Xianliang, Zhihao Liu, Wei Jiang, Chuheng Zhang, Li Zhao, Lei Song, and Jiang Bian. "A Versatile Multi-Agent Reinforcement Learning Benchmark for Inventory Management," 2023. https://doi.org/10.48550/ARXIV.2306.07542.
11. Rolf, Benjamin, Ilya Jackson, Marcel Müller, Sebastian Lang, Tobias Reggelin, and Dmitry Ivanov. "A Review on Reinforcement Learning Algorithms and Applications in Supply Chain Management." *International Journal of Production Research* 61, no. 20 (October 18, 2023): 7151–79. https://doi.org/10.1080/00207543.2022.2140221.
12. Dhaliwal, Amandeep. "The Rise of Automation and Robotics in Warehouse Management," 63–72, 2020. https://doi.org/10.1201/9781003032410-5.
13. Vivaldini, K., Jorge Galdames, Thales Pasqual, Roberto Araujo, Rafael Sobral, Marcelo Becker, and Glauco Caurin. "*Robotic Forklifts for Intelligent Warehouses: Routing, Path Planning, and Auto-Localization*". *Proceedings of the IEEE International Conference on Industrial Technology*, 2010. https://doi.org/10.1109/ICIT.2010.5472487.
14. Pinkam, Nantawat, Francois Bonnet, and Nak Chong. *Robot Collaboration in Warehouse*, 2016. https://doi.org/10.1109/ICCAS.2016.7832331.
15. Rajana, Sandesh. *Robotics for the Supply Chain*, 2018. https://doi.org/10.13140/RG.2.2.28081.43361.

16. Oliveira, George S., Juha Röning, Patricia D. M. Plentz, and Jônata T. Carvalho. “Efficient Task Allocation in Smart Warehouses with Multi-Delivery Stations and Heterogeneous Robots.” arXiv, February 28, 2022. https://doi.org/10.48550/arXiv.2203.00119.
17. Sultana, Nazneen N., Hardik Meisheri, Vinita Baniwal, Somjit Nath, Balaraman Ravindran, and H. Khadilkar. “Reinforcement Learning for Multi-Product Multi-Node Inventory Management in Supply Chains.” *ArXiv*, June 7, 2020. https://www.semanticscholar.org/paper/51408188c013b80faf38a69dc15f729ce2b9ac9e.
18. Cardoso, Rafael C., and Angelo Ferrando. “A Review of Agent-Based Programming for Multi-Agent Systems.” *Computers* 10, no. 2 (February 2021): 16. https://doi.org/10.3390/computers10020016.
19. Shiflet, Angela B., and George W. Shiflet. “An Introduction to Agent-Based Modeling for Undergraduates.” *Procedia Computer Science* 29 (2014): 1392–1402. https://doi.org/10.1016/j.procs.2014.05.126.
20. Kardos, Csaba, Catherine Laflamme, Viola Gallina, and Wilfried Sihn. “Dynamic Scheduling in a Job-Shop Production System with Reinforcement Learning.” *Procedia CIRP* 97 (January 1, 2021): 104–9. https://doi.org/10.1016/j.procir.2020.05.210.
21. Zhang, Ming, Yang Lu, Youxi Hu, Nasser Amaitik, and Yuchun Xu. “Dynamic Scheduling Method for Job-Shop Manufacturing Systems by Deep Reinforcement Learning with Proximal Policy Optimization.” *Sustainability* 14, no. 9 (January 2022): 5177. https://doi.org/10.3390/su14095177.
22. Yoon, Sukmin, and Jinwhan Kim. “Efficient Multi-Agent Task Allocation for Collaborative Route Planning with Multiple Unmanned Vehicles.” *IFAC-PapersOnLine*, 20th IFAC World Congress, 50, no. 1 (July 1, 2017): 3580–85. https://doi.org/10.1016/j.ifacol.2017.08.686.
23. A, Ramaa., K. N. Subramanya, and T. M. Rangaswamy. “Impact of Warehouse Management System in a Supply Chain.” *International Journal of Computer Applications* 54, no. 1 (September 25, 2012): 14–20. https://doi.org/10.5120/8530-2062.
24. Qie, Han, Dianxi Shi, Tianlong Shen, Xinhai Xu, Yuan Li, and Liujing Wang. “Joint Optimization of Multi-UAV Target Assignment and Path Planning Based on Multi-Agent Reinforcement Learning.” *IEEE Access* 7 (2019): 146264–72. https://doi.org/10.1109/ACCESS.2019.2943253.
25. Li, Jiaoyang, Andrew Tinka, Scott Kiesel, Joseph W. Durham, T. K. Satish Kumar, and Sven Koenig. “Lifelong Multi-Agent Path Finding in Large-Scale Warehouses.” arXiv, March 12, 2021. http://arxiv.org/abs/2005.07371.
26. Zhang, Jie. *Multi-Agent Based Production Planning and Control*. First edition. Hoboken, NJ, USA: John Wiley & Sons, Inc, 2017.
27. Shyalika, Chathurangi, Thushari Silva, and Asoka Karunananda. “Reinforcement Learning in Dynamic Task Scheduling: A Review.” *SN Computer Science* 1, no. 6 (November 2020): 306. https://doi.org/10.1007/s42979-020-00326-5.
28. Liu, Minghua, Hang Ma, Jiaoyang Li, and Sven Koenig. “Task and Path Planning for Multi-Agent Pickup and Delivery.” In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, 1152–60. AAMAS ’19. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, 2019.
29. Gronauer, Sven, and Klaus Diepold. “Multi-Agent Deep Reinforcement Learning: A Survey.” *Artificial Intelligence Review* 55, no. 2 (February 1, 2022): 895–943. https://doi.org/10.1007/s10462-021-09996-w.
30. Dhaliwal, Amandeep. “12. The Rise of Automation and Robotics in Warehouse Management,” 63–72, 2020. https://doi.org/10.1201/9781003032410-5.

31. Vivaldini, K., Jorge Galdames, Thales Pasqual, Roberto Araujo, Rafael Sobral, Marcelo Becker, and Glauco Caurin. *13. Robotic Forklifts for Intelligent Warehouses: Routing, Path Planning, and Auto-Localization*. *Proceedings of the IEEE International Conference on Industrial Technology*, 2010. https://doi.org/10.1109/ICIT.2010.5472487.
32. Zhu, Anmin, and Simon Yang. "A Neural Network Approach to Dynamic Task Assignment of Multirobots." *Neural Networks, IEEE Transactions On* 17 (October 1, 2006): 1278–87. https://doi.org/10.1109/TNN.2006.875994.
33. Ma, Hang, and Sven Koenig. "Optimal Target Assignment and Path Finding for Teams of Agents." arXiv, December 16, 2016. https://doi.org/10.48550/arXiv.1612.05693.
34. Macalpine, Patrick, Francisco Barrera, and Peter Stone. "Positioning to Win: A Dynamic Role Assignment and Formation Positioning System." *AAAI Workshop - Technical Report*, January 1, 2012. https://doi.org/10.1007/978-3-642-39250-4_18.
35. Park, Junyoung, Sanjar Bakhtiyar, and Jinkyoo Park. "ScheduleNet: Learn to Solve Multi-Agent Scheduling Problems with Reinforcement Learning." arXiv, June 6, 2021. https://doi.org/10.48550/arXiv.2106.03051.
36. X. Zhang and P. Wang, "Grid-Map-Based Path Planning and Task Assignment for Multi-Type AGVs in a Distribution Warehouse," *Mathematics*, vol. 11, no. 13, p. 2802, 2023. https://www.mdpi.com/2227-7390/11/13/2802 .
37. O. Kulak, et al., "Collaborative Optimization of Task Scheduling and Multi-Agent Path Planning in Automated Warehouses," *Complex & Intelligent Systems*, 2023. https://link.springer.com/article/10.1007/s40747-023-00719-1.
38. V. Lesch, P. B. M. Müller, M. Krämer, S. Kounev, and C. Krupitzer, "A Case Study on Optimization of Warehouses," *arXiv preprint arXiv:2112.12058*, 2022. https://arxiv.org/abs/2112.12058.
39. G. K. Adil, "Efficient Formation of Storage Classes for Warehouse Storage Location Assignment: A Simulated Annealing Approach," *Omega*, vol. 36, no. 4, pp. 609-618, 2008. https://www.sciencedirect.com/science/article/abs/pii/S030504830700076X.
40. G. Marchetti, et al., "Analytical Model to Estimate Performances of Autonomous Vehicle Storage and Retrieval Systems for Product Totes," *International Journal of Production Research*, vol. 50, no. 24, pp. 7134-7148, 2012. https://www.tandfonline.com/doi/abs/10.1080/00207543.2011.639815.

APPENDIX A

An Introduction to Multi-Agent Systems

Multi-Agent Systems (MAS) are a captivating subfield of distributed artificial intelligence. These systems encompass multiple autonomous or semi-autonomous software entities, termed agents, that interact within a shared environment. Agents possess their own goals, problem-solving capabilities, and decision-making processes. In a multi-agent system, the agents collaborate, coordinate, and sometimes compete to achieve individual and collective objectives.

MAS offer unique advantages for modeling and solving complex problems that would prove difficult or intractable with a traditional, centralized approach. They promote decentralization, flexibility, and adaptability, making them a potent tool in tackling challenges across diverse domains.

Agent Architectures

Agent architectures define the fundamental blueprints that dictate how agents within a multi-agent system perceive the environment, reason, and act. Here's a breakdown of the primary categories:

Reactive Agents

Core Principle: Direct stimulus-response mapping. Agents react to changes in the environment based on predetermined rules or behaviors.

Characteristics:

- Fast response times
- Simple in design
- Limited decision-making capabilities
- No internal model of the world

Examples: Thermostat controlling temperature, basic obstacle avoidance robots.

Deliberative Agents

Core Principle: Symbolic reasoning and planning. Agents maintain an internal symbolic representation (a model) of the world, which they use for logical reasoning to create plans and make informed decisions.

Characteristics:

- More complex decision-making
- Can handle complex goals and plans

- Can reason about their beliefs, desires, and intentions (BDI architecture is a common example).
- May be slower to respond than reactive agents due to deliberation.

Examples: Chess-playing AI, sophisticated planning systems.

Hybrid Agents

Core Principle: A blend of reactive and deliberative components. Agents have both a reactive layer for quick responses and a deliberative layer for more complex problem-solving and planning.

Characteristics:

- Balance adaptability and complex decision-making
- More flexible than purely reactive or deliberative agents

Examples: Self-driving cars (reactive for immediate actions, deliberative for route planning), complex robots operating in dynamic environments.

Communication and Coordination

In MAS, communication and coordination are the cornerstones that allow agents to work together effectively. They go hand-in-hand:

Communication: The exchange of information between agents. This can include sharing observations, goals, intentions, plans, and negotiated agreements.

Coordination: The process of agents aligning their actions, plans, and resource usage to achieve shared objectives while avoiding conflicts.

To facilitate communication, agents use specialized communication languages and protocols. These provide structure and syntax for interactions, such as KQML (Knowledge Query and Manipulation Language) and FIPA ACL (Foundation for Intelligent Physical Agents- Agent Communication Language). Ontologies are also important, as they offer a shared vocabulary ensuring that agents interpret exchanged information in the same way. Beyond the languages themselves, the medium of communication is important – agents might interact via direct messaging, broadcast messaging, or even indirect communication through modifications of their environment.

Coordination in MAS can be either explicit or implicit. Explicit coordination relies on direct agent-to-agent communication for negotiation and conflict resolution. In contrast, implicit coordination involves agents modifying their behavior in reaction to environmental cues and the actions of others, without the need for direct negotiation. Coordination mechanisms range from joint planning approaches to market-based systems that use auctions, bids, and pricing. Negotiation

protocols and consensus algorithms are also vital tools for resolving conflicts, finding agreements, and distributing tasks within a multi-agent system.

Effective communication and coordination in MAS can be a real challenge. Limitations in the amount of information that can be transmitted, uncertainty about the environment or other agents, and the constantly changing conditions within dynamic environments all complicate matters. The difficulty of maintaining efficient coordination also increases as the number of agents in the system grows.

Cooperation and Negotiation

Cooperation and negotiation lie at the heart of effective Multi-Agent Systems (MAS). Cooperation involves agents working together to achieve shared goals that may be beyond the abilities of any single agent. Negotiation comes into play when agents have individual goals and preferences that might conflict, requiring them to find mutually agreeable solutions.

One crucial aspect of cooperation is task decomposition and allocation. This involves breaking down a complex problem into smaller tasks and efficiently assigning those tasks to the most suitable agents within the system. Effective strategies are needed to optimize this process for the best overall outcome.

Negotiation techniques are essential for resolving conflicts and reaching mutually beneficial agreements in a MAS. Formal negotiation protocols provide structured ways for agents to propose offers, counteroffers, and ultimately find compromises. These protocols facilitate resource exchange, conflict resolution, and general cooperation within the system.

Negotiation in MAS faces several challenges. Agents may have limited information about the preferences or capabilities of others, and they might need to operate in dynamic environments where goals and conditions shift. Moreover, self-interested agents might pursue their own benefits at the expense of overall system goals. Designing negotiation mechanisms that incentivize cooperation while ensuring fairness and efficiency is an ongoing research area.

Distributed Problem Solving

Distributed Problem Solving (DPS) is a core paradigm within Multi-Agent Systems. It deals with how to decompose a complex problem into smaller, interrelated subproblems that can be tackled by a collection of distributed agents. Each agent holds expertise in a specific domain or controls certain resources, contributing to finding an overall solution.

One key technique within DPS is distributed constraint satisfaction. This involves constraints that span multiple agents, such as ensuring that two agents don't choose conflicting time slots for a shared resource. The goal is for agents to coordinate their choices in a way that satisfies all constraints within the system.

Planning also plays a vital role in distributed problem solving within MAS. Agents must often coordinate their individual plans to accommodate interdependencies and potential conflicts. This

might involve generating a centralized plan, but a more distributed approach allows for greater flexibility and robustness. Joint planning, coordination during plan execution, and techniques to resolve conflicts as they arise are essential within this domain.

Distributed problem-solving in MAS offers several advantages. Decentralization makes systems more robust, as a single agent's failure won't cripple the entire system. It also allows for exploiting specialized agents, leveraging their unique knowledge and capabilities to tackle specific subproblems. Furthermore, DPS can improve solution time, as tasks are solved concurrently by multiple agents.

An Introduction to Reinforcement Learning

Reinforcement Learning (RL) is a powerful machine learning paradigm that focuses on how intelligent agents can learn to make optimal decisions in complex environments. Unlike supervised learning, where you provide labeled data, or unsupervised learning, where the goal is to uncover patterns, RL emphasizes learning through trial-and-error interactions.

The RL framework involves an agent situated in an environment. The agent performs actions, which affect the state of the environment. In turn, the environment provides rewards (or penalties) that give feedback on the agent's actions. The key goal of RL is to train the agent to learn a policy (a strategy or decision-making rule) that maximizes its cumulative rewards over time.

Markov Decision Processes (MDPs)

Markov Decision Processes provide a structured way to think about sequential decision-making problems where the outcomes are partially random and partially under the control of the agent. The core idea behind an MDP is that the current state of the environment encapsulates all the information needed to make the next "best" decision, regardless of the full history that led to that state. This is called the Markov Property.

Components of an MDP

- States (S): The set of possible situations the agent can be in within the environment.
- Actions (A): The set of choices the agent can make at any given state.
- Transition Probabilities (P): The probabilities of moving from one state to another based on a specific action (e.g., if I take action A in state S, what's the probability I'll end up in state S'?).
- Rewards (R): The immediate feedback signal received by the agent for taking an action in a given state.
- Discount Factor (γ): A value between 0 and 1 that determines how much the agent cares about immediate rewards vs. long-term cumulative rewards.

Why MDPs are Essential for Understanding RL

1. Formalizing the RL Problem: MDPs provide the mathematical language to describe the core elements of a reinforcement learning environment – states, actions, rewards, etc. Without this formalization, it's challenging to rigorously define RL problems and develop algorithms to solve them.
2. Foundation for Value and Policy: Key RL concepts like value functions (how good is it to be in a state) and policies (what action should I take in a state) are directly grounded in the MDP framework. Algorithms like Q-learning and Policy Gradients operate on the dynamics and rewards described by the MDP.
3. Understanding Optimality: MDPs allow us to define the concept of "optimal behavior". The goal in RL is to find a policy that maximizes the expected sum of discounted reward – finding these optimal policies often depends on carefully considering the state transition dynamics of the MDP.

Value Functions

Value functions are the heart of many reinforcement learning algorithms. They provide a way to assess the long-term desirability of states or state-action pairs. There are two main types:

- State-value function (V(s)): This function estimates the expected total discounted reward an agent can receive if it starts in a particular state 's' and then follows a given policy. It tells you how "valuable" being in a certain state is in terms of future potential rewards.
- Action-value function (Q (s, a)): This function estimates the expected total discounted reward if an agent starts in state 's', takes action 'a', and then follows a given policy. It tells you the "value" of taking a specific action in a given state under a policy.

Why Value Functions are Important

1. Decision Making: Value functions guide the agent's decision-making process. An agent can learn to take actions that lead to states with higher values, increasing its chances of maximizing long-term reward.
2. Policy Evaluation: Given a policy, value functions help evaluate how good that policy is. This is essential for policy-based methods and for comparing different policies.
3. Learning without a Model: Value-based RL algorithms like Q-learning allow the agent to learn the optimal policy without explicitly needing a model of the environment. They estimate the value functions directly from experience, and those values indirectly guide action choices.

Bellman Equation

Bellman equations express the core relationships between value functions in Reinforcement Learning. They break down the value of a state (or state-action pair) into two components:

1. The immediate reward: The reward received by the agent for being in the current state (or taking a specific action in that state).
2. The discounted value of the next state: The future expected reward the agent can receive from the next possible state, discounted by a factor (γ) that emphasizes immediate rewards over long-term ones.

Types of Bellman Equations

There are two main types of Bellman equations:

1. Bellman Expectation Equation (for state-value functions): It expresses the value of a state as the expected reward for being in that state, plus the expected discounted value of the next possible states.
2. Bellman Optimality Equation (for optimal state-value functions): It describes the value of a state under the optimal policy. Here, instead of simply averaging the value of successor states, it takes the maximum possible value from the next states, as an optimal policy would select the best action leading to the highest value.

Why Bellman Equations are Fundamental to RL

- Foundation for Algorithms: Many popular RL algorithms like Q-learning, SARSA, and Value Iteration are directly derived from Bellman equations. They provide the recursive structure that allows the agent to update its estimates of value functions based on experience.
- The Principle of Optimality: Bellman equations embody the principle of optimality: that optimal policies have the property that no matter what the initial state and first action are, the remaining actions must constitute an optimal policy with regard to the state resulting from that first action.

Policy Based Methods

Policy-based methods directly learn the optimal policy, which maps states to the best actions. The policy is represented as a function (often parameterized by a neural network). These methods aim to optimize this policy function itself, typically using methods like policy gradients.

- Pros: Can handle continuous action spaces gracefully, may converge to a better policy in some environments, and can learn stochastic policies (where the best action isn't always deterministic).

- Cons: Can suffer from high variance (fluctuations in learning), may be less sample-efficient, and can have a tendency to get stuck in local optima (suboptimal policies).

Value-Based Methods

Value-based methods focus on learning value functions (state-value or action-value functions). They estimate how good it is to be in a certain state or how good it is to take a specific action in a state. An optimal policy is then implicitly derived by choosing actions that lead to states with the highest values. Popular methods include Q-Learning and SARSA.

- Pros: Tend to be more stable and sample-efficient, easier to analyze theoretically for convergence. Better for environments with discrete action spaces.
- Cons: Struggle with continuous action spaces since you need to compare values for all possible actions, and generally can only learn deterministic policies.

Hybrid Methods

Actor-Critic methods offer a blend of the two approaches. They have an 'actor' (the policy function, like in policy-based methods) and a 'critic' (which learns a value function, like in value-based methods). The critic guides the update of the actor's policy, potentially combining the benefits of both approaches.

An Analogy for Policy and Value Based Methods

It's a bit like choosing directions on a map. Policy-based methods are like having a compass that directly points you towards the best direction. Value-based methods are like having a map with elevation contours, allowing you to see the entire landscape and choose a trajectory based on the terrain.

Exploration vs Exploitation

At the heart of reinforcement learning lies a fundamental dilemma: should the agent stick with actions that it knows lead to favorable results (exploitation), or should it try out new, potentially suboptimal actions in the hope of discovering something even better (exploration)? An agent that exploits perfectly may find itself stuck in a suboptimal way of behaving, while an agent that only explores might never converge to a good solution.

Why is this trade-off important?

- Avoiding Plateaus: Early in learning, an agent doesn't have a lot of knowledge about its environment. Pure exploitation would lead to premature convergence on a potentially mediocre policy. Exploration can help discover better strategies.

- Long-term Rewards: Exploration is crucial for discovering strategies that may initially seem unrewarding but lead to much higher long-term payoffs. Think of it as the RL agent taking some calculated risks for the potential of a big reward.
- Dynamic Environments: Even if an agent finds a good strategy, environments can change. Continuous exploration allows for adaptation to these changes, ensuring the agent doesn't fall behind if circumstances shift.

Techniques for Balancing Exploration and Exploitation

- Epsilon-Greedy: A simple method where the agent chooses a random action with probability epsilon and takes the "best" known action otherwise.
- Boltzmann Exploration: Actions are chosen with probabilities based on their estimated Q-values, encouraging the selection of good actions, but allowing for less optimal actions occasionally.
- Upper Confidence Bound (UCB): Actions are selected to balance maximizing reward and reducing uncertainty about the value of less-explored options.
- Information-Based Exploration: Actions are chosen to maximize the gain in knowledge (information) about the environment.

Model-Based Learning vs Model-Free Learning

In model-based RL, the agent attempts to explicitly learn a model of its environment. This model describes how the environment reacts to the agent's actions: specifically, it predicts the next state and reward given the current state and an action. With this model, the agent can plan its actions by reasoning about potential future scenarios before actually taking them.

- Pros: Can be extremely sample efficient, as the agent can generate additional data through its internal model. This is useful when real-world interactions are expensive or slow. Additionally, model-based methods can be better for quickly adapting to changes in the environment dynamics.
- Cons: Accuracy heavily depends on how well the learned model matches the true environment. If the environment is complex or the model is imperfect, performance suffers. Also, planning with the model adds computational overhead.

Model-free RL algorithms don't try to build an explicit model of how the environment works. Instead, they learn directly from experience. The agent interacts with the environment, observes the rewards and transitions, and updates its policy or value functions without ever forming a representation of how the environment works "behind the scenes." Popular approaches include Q-learning, SARSA, and policy-gradient methods.

- Pros: Can be more robust to imperfect modeling assumptions. Often simpler to implement than model-based methods. Can be highly effective in environments where an accurate model is hard to learn.

- Cons: Generally, sample inefficient, requiring a large amount of real-world interaction data to learn effectively. Can also be less adaptable if the reward function or environment dynamics change significantly.

An Analogy

In essence, a model-based agent is like a chess player who carefully considers multiple potential moves and their consequences before taking an action. A model-free agent is more like a player who learns through trial and error, gradually discovering winning strategies through experience.

APPENDIX B

The Code for Model Creation

```
# Create the DQN model
def create_dqn(tasks_shape, devices_shape, num_actions):
    print('\n==>[creating DQN model]: tasks_shape:', tasks_shape, 'devices_shape:', \
          devices_shape, 'num_actions:', num_actions, '\n')

    # Define Input Shapes (Adjust based on your exact features)
    task_input_shape = tasks_shape  # 2D tensor (num_tasks, 9). 9 means number of feat
    device_input_shape = devices_shape  # 2D tensor (num_devices, 4). 4 means number o

    # Define Input Layers
    task_input = Input(shape=task_input_shape) #
    device_input = Input(shape=device_input_shape)


    task_x = Dense(10, activation='relu')(task_input)  #  Initial Task Processing
    task_x = Dense(100, activation='relu')(task_x)

    device_x = Dense(16, activation='relu')(device_input)  # Initial Device Processing
    combined = Concatenate()([task_x, device_x]) # Combine processed Information
    x = Dense(32, activation='relu')(combined) #Combined Decision Layers
    x1 = Dense(100, activation='relu')(x)  # # One layer after the combined decision l

    flattened_x1 = Reshape((1, 10 * num_actions))(x1)  # Flatten the input for the out
    task_output = Dense(num_actions, activation='linear')(flattened_x1)  # Output for

    model = Model(inputs=[task_input], outputs=[task_output]) # Instantiate the Model

    # Compile the Model
    model.compile(optimizer=keras.optimizers.Adam(),
                  loss='mean_squared_error',
                  metrics=['accuracy', 'mean_squared_error'])

    return model
```

Figure Appendix B.1 – DQN Model Implementation

Description

The code defines a function called “create_dqn()” that takes three inputs:

- tasks_shape: The shape of the task input data.

- devices_shape: The shape of the device input data.
- num_actions: The number of possible actions that the agent can take.

The function first defines the input shapes for the task and device data. Then, it creates two input layers, one for the task data and one for the device data. The task data is passed through two dense layers with relu activation functions. The device data is passed through a single dense layer with a relu activation function. The outputs of the two dense layers are then concatenated together.

The concatenated output is then passed through two more dense layers with relu activation functions. The output of the final dense layer is reshaped into a one-dimensional vector and then passed through a dense layer with a linear activation function. This final dense layer has an output size equal to the number of possible actions.

The function then creates a Keras model that takes the task input as input and the output of the final dense layer as output. The model is compiled using the Adam optimizer and the mean squared error loss function.

Overall, the code creates a DQN model that can be used to make decisions in an RL environment. The model takes task and device data as input and outputs a probability distribution over the possible actions that the agent can take. The model can be trained using reinforcement learning techniques to learn how to make decisions that will maximize the agent's reward.

The Reshape layer is used to reshape the output of the second dense layer into a one-dimensional vector. This is necessary because the output layer of a DQN model must be a one-dimensional vector with a size equal to the number of possible actions.

The “mean_squared_error” loss function is commonly used in DQN models. It measures the difference between the predicted Q-values and the target Q-values.

The Adam optimizer is a popular optimizer for training neural networks. It is an adaptive learning rate optimizer that adjusts the learning rate for each parameter based on the observed gradients.