UNIVERSITY OF MODENA AND REGGIO EMILIA
Department of Sciences and Methods for Engineering
Doctorate School in Industrial Innovation Engineering
XXXVI Cycle

Optimization of dynamic transportation systems

DOCTORAL THESIS

Author:
Francesco GALLESI

Supervisor:
Prof. Manuel IORI

Coordinator:
Prof. Franco ZAMBONELLI

Academic year 2022/2023

UNIVERSITY OF MODENA AND REGGIO EMILIA

Abstract

Doctorate School in Industrial Innovation Engineering
Department of Sciences and Methods for Engineering

Doctor of Philosophy

Optimization of dynamic transportation systems

by Francesco GALLESI

This doctoral thesis explores optimization problems in the context of transportation and mobility systems. The study focuses on two specific topics: recent advances on pickup and delivery problems in Automated Guided Vehicles (AGVs) systems, and route planning for a fleet of vehicles in urgent delivery systems. Regarding the first topic, the research involves a detailed review of the surveys in the literature on scheduling AGVs, a mathematical model to formalize the problem and a collection of the challenges and opportunities in the context of scheduling AGVs and its variants. Then, the first work is expanded with a collection of the recent advances related to AGVs systems in general, including mathematical models for pickup and delivery problem with battery management and multi-load variants, and promising future directions in this topic. The work on the second topic proposes a novel branch-and-regret algorithm to solve the problem of urgent deliveries, known in the literature as *Same-Day Delivery Problem*. The aim is to maximize the served requests thanks to an algorithm able to incorporate sampled scenarios to anticipate future events and make informed routing decisions. The computational results show the performance of the algorithm, evidencing the superiority of the proposed branch-and-regret compared with state-of-the-art algorithms from the literature. In conclusion, this doctorate thesis makes significant contributions to the advancement of operations research in the field of transportation, providing new prospects and innovative solutions.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In the current economic and technological context, efficient transportation management plays a fundamental role for companies operating in sectors such as logistics, intralogistics, and e-commerce. Operations research, as a scientific discipline dedicated to modeling, analyzing, and solving complex problems, provides tools and methodologies to address the challenges associated with transportation.

This doctoral thesis focuses on the application of operations research to dynamic transportation problems, with a specific emphasis on two topics. The first topic concerns the transportation of goods utilizing Automated Guided Vehicles (AGVs) in the field of intralogistics (an example of AGVs is provided in Figure 1.1). The increasing automation of facilities and the need for greater efficiency in the movement of goods have made AGVs a promising avenue for optimizing internal operations within companies (see, e.g., Fazlollahtabar and Saidi-Mehrabad [33], De Ryck, Versteyhe, and Debrouwere [29], Rashidi, Matinfar, and Parand [77]). The present study is motivated by the considerable interest within the scientific community regarding the coordination of a fleet of AGVs, as evidenced by the rapid growth in the number of publications dedicated to this area in recent years. Another driving factor behind this research stems from the interest cultivated throughout my industrial doctorate at E80 Group [32], a leading company in the AGV market, which provides an industrial perspective on the problem at hand. E80 Group is an Italian-based organization situated in Viano, Italy, specialized in the development of automated and integrated intralogistics solutions for consumer goods manufacturers in sectors such as beverages, food, and tissue. Within E80 Group, my role as a Research and Development (R&D) engineer entails overseeing traffic management and optimization in AGV systems. Specifically, during the course of my doctoral studies, my focus has been on addressing the order scheduling problem for AGVs in highly dynamic environments characterized by rapidly changing information, necessitating the recomputation of solutions every few seconds. The collaborative efforts pursued during my industrial doctorate at E80 Group have provided a unique opportunity to extensively explore this subject matter through comprehensive literature research, which is presented in Chapters 2 and 3 of this thesis.

The second topic addressed in this thesis concerns the urgent delivery of products with requests in the same day. This regards real world application as such the e-commerce. In an increasingly widespread and competitive online purchasing context, the speed and efficiency of product delivery have become key factors in ensuring customer satisfaction and loyalty to a particular online store. Consumers expect to receive their purchases in the shortest possible time, often within the same day the order was placed. This type of service offers numerous advantages for both consumers and businesses. For customers, same-day delivery allows them to immediately receive what they need, avoiding prolonged waits and enhancing the overall purchasing experience. Moreover, it is particularly beneficial for urgent purchases or

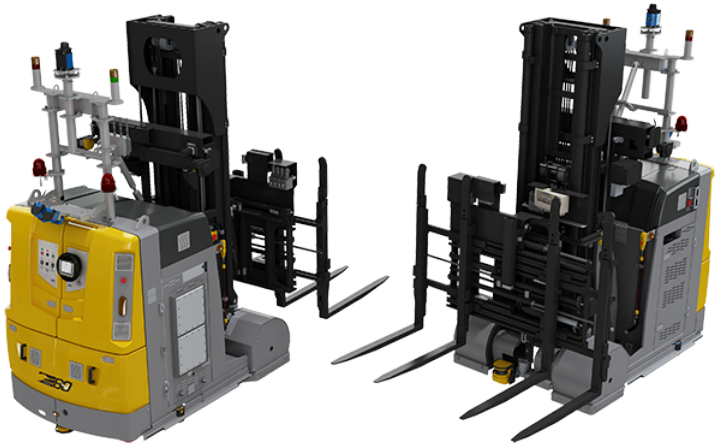


Figure 1.1: Example of Automated Guided Vehicles

situations where customers require rapid product delivery. From a business perspective, same-day delivery systems represent a distinguishing element that can confer a competitive advantage. Providing a fast and reliable delivery service helps attract and retain loyal customers, increasing the likelihood of repeat business. Additionally, timely delivery can contribute to reducing costs associated with potential returns or customer complaints. To enable same-day delivery, innovative strategies and solutions in the fields of logistics and optimization are necessary. Sophisticated order management, routing, and shipment tracking systems become essential for ensuring efficient planning and swift delivery. The utilization of advanced algorithms, as in the case of the same-day delivery problem (see Voccia, Campbell, and Thomas [94]), allows for resource allocation optimization, optimal route planning, and effective management of unforeseen events that may occur along the way. In conclusion, same-day delivery systems play an increasingly significant role in the e-commerce sector, meeting customer expectations and contributing to business competitiveness. Continuous research and development in the field of operations and optimization of delivery systems are essential for improving efficiency, reliability, and timeliness of shipments, thereby offering an increasingly satisfactory and high-quality purchasing experience. Chapter 4 presents the research conducted to devise an innovative algorithm for managing same-day deliveries.

In addition to the works reported in Chapters 2, 3 and 4, another significant project was undertaken during this industrial doctorate in collaboration with E80 Group. Following the literature review presented in Chapter 2, its extensions in Chapter 3 and the theoretical work on the same-day delivery problem discussed in Chapter 4, this thesis extends its focus to a real-world application within the context of E80 Group. The project centers around the development of an optimization algorithm applied to traffic management in an AGV system. Specifically, my research revolves around the scheduling of orders with the objective of maximizing system throughput while minimizing delays. The algorithm needs to work for large-dimensional AGV system (up to 200 AGVs in a single plant) in highly dynamic environments, where information changes quickly and the solution needs to be recomputed every few seconds. In addition, it is essential for the product to be versatile and not customized, capable of operating on diverse systems, and adaptable to varying customer demands. The project encompasses a comprehensive study

of the problem, and the development of an algorithm aimed at enhancing the existing system employed by E80 Group. It is worth noting that the algorithm has been successfully implemented and is currently operational in four customer plants of E80 Group. This work cannot be published because of the non-disclosure policy of the company, and, for this reason, I do not report other details regarding this major undertaking of my doctoral research. Nonetheless, it serves as a testament to the practical applicability and impact of the research conducted during the industrial doctorate program. The successful implementation of the developed algorithm in real-world settings demonstrates the relevance and effectiveness of operations research and optimization techniques in addressing complex transportation problems within industry contexts.

By undertaking the analysis and resolution of the specific problems reported, this thesis aims to contribute to the advancement of knowledge in the field of operations research applied to transportation. The outcomes obtained will offer fresh perspectives and practical solutions for companies operating within the logistics sector, thereby facilitating the optimization of their operations and overall efficiency.

More in detail, the thesis is organized as follows. Chapter 2 presents a comprehensive survey on the scheduling problem in AGV systems. The chapter begins with a detailed review of existing surveys available in the literature, summarizing the key findings and methodologies employed in previous research. This initial part serves as a foundation for understanding the current state of knowledge and identifying gaps that need to be addressed. Following the survey, the chapter proceeds to propose a mathematical model for formally defining the scheduling problem in AGV systems. While the model itself may not be innovative or groundbreaking, its purpose is to establish a clear and concise representation of the problem, providing a basis for further analysis and development. In addition to the model, the chapter explores open challenges and future directions highlighted in various articles related to scheduling and its variations within AGV systems. This includes addressing issues such as battery management, multi-load considerations, and integration with other related systems. By examining these challenges, the chapter aims to identify opportunities for further research and potential areas of improvement in AGV scheduling.

An extension of the survey conducted in Chapter 2 is presented in Chapter 3, wherein additional aspects of coordinating a fleet of AGVs within an intralogistics system are explored. The chapter presents recent developments related to AGV systems, with a specific focus on the scheduling problem. The chapter begins with a collection of books and surveys available in the literature, forming a crucial state-of-the-art overview of the topic. It then introduces a mathematical model that extends the general model proposed in Chapter 2. This extended model takes into consideration variations in the Pickup and Delivery Problem (PDP), specifically addressing battery management and multi-load scenarios. The battery management aspect accounts for the time required to charge or replace AGV batteries, while the multi-load variant involves the loading of multiple items before executing a drop operation. Subsequently, the chapter examines challenges and opportunities outlined in various articles discussing recent advancements across different aspects of the topic, including scheduling, path planning, conflict avoidance, and integrated production with AGVs.

Chapter 4 of the thesis presents an innovative algorithm based on the branch-and-regret approach (see Hvattum, Løkketangen, and Laporte [47]) to address the same-day delivery problem. The chapter begins by introducing the problem and

highlighting its significance in the context of time-constrained deliveries, particularly in the e-commerce industry. The proposed algorithm is designed to optimize routing plans and maximize the number of served requests while minimizing the traveled distance. It utilizes a branch-and-regret framework, which incorporates sampled scenarios to anticipate future events and make informed routing decisions. The algorithm also employs an adaptive large neighborhood search (see Ropke and Pisinger [79]) to iteratively improve the routing plans. To evaluate the performance of the proposed algorithm, extensive computational experiments are conducted on a wide range of instances. The chapter includes a detailed comparison with the most influential works in the literature (Voccia, Campbell, and Thomas [94], Tirado et al. [87], Ulmer, Thomas, and Mattfeld [92]) on the same-day delivery problem. This comparison examines various performance metrics such as the number of served requests, traveled distance, and computational effort, demonstrating the effectiveness and superiority of the proposed branch-and-regret algorithm.

Finally, concluding remarks are reported in Chapter 5.

Chapter 2

Scheduling Automated Guided Vehicles: challenges and opportunities *

Automated Guided Vehicles (AGVs) play a fundamental role in different logistic systems, being widely used for the automatic handling of materials, goods, and containers. The management of AGVs requires the solution of several optimization problems, such as task allocation/scheduling, routing, and path planning, which are often enriched by additional attributes, such as multi-load, battery constraints, and conflict avoidance. Many of these problems are faced in the real-world context of the Italian company E80 Group, one of the world leaders in the production of AGV systems. The literature is huge for all the aforementioned problems, and hence we focus only on the problem of scheduling AGVs, modeled as a Pickup and Delivery Problem (PDP). In particular, we propose a PDP formulation, discuss real-world and literature scheduling applications, and indicate challenges and research opportunities providing a guide for future researches.

2.1 Introduction

In the last decades, AGVs have become a common equipment to transport materials, goods and containers in logistic systems, changing the work habits of many companies. The AGVs provide efficient and flexible solutions for transportation and manufacturing systems, and their management has consequently become a crucial logistic activity. The interest towards AGVs and the optimization problems arising from their use is testified by the rapid increase in the number of works devoted to these topics (see Figure 2.1).

In particular, as defined by De Ryck, Versteyhe, and Debrouwere [29], there are five main problems related to AGVs: task allocation/scheduling, localization, path planning, motion planning, and vehicle management for AGV coordination. More precisely, task allocation/scheduling considers assigning a set of tasks to the AGVs; localization refers to the position of each AGV in the environment; path planning defines the shortest path to the destination; motion planning avoids collisions with other AGVs and static objects; and vehicle management considers the status of the AGVs, focusing on the errors and the maintenance. Some variants of these problems

*The results of this chapter were accepted as: Gallesi, F., Praxedes, R., Iori, M., Locatelli, M., and Subramanian, A. "Scheduling automated guided vehicles: challenges and opportunities". In *Proceeding of the International Conference on Optimization and Decision Science 2023, AIRO Springer Series* (2023, forthcoming).

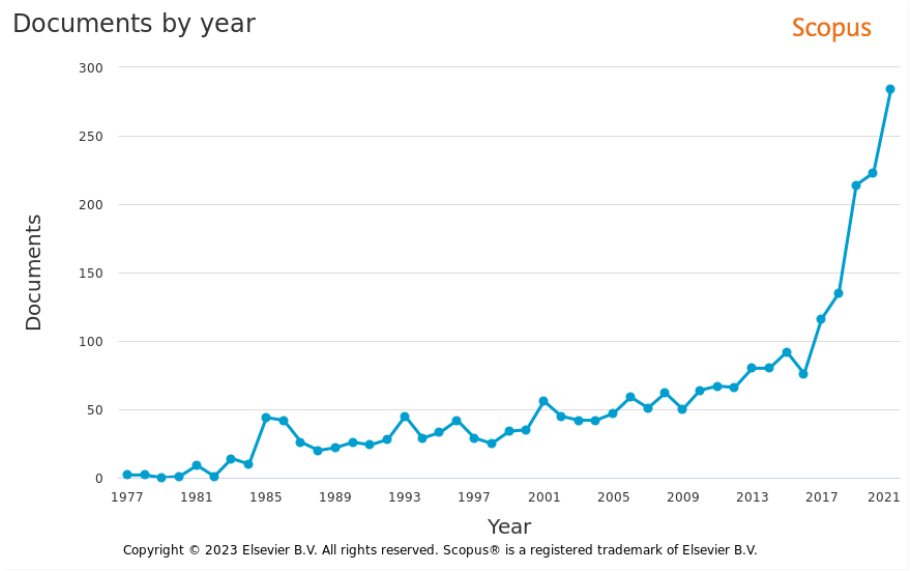


Figure 2.1: Number of publications per year that include the keywords "AGVs", "AGV", "Automated Guided Vehicles", "AGV systems" or "AGVs systems" in their titles.

consider additional attributes and environmental aspects, such as multi-load, battery constraints, and conflict-free routing, just to mention a few. Multi-load scheduling includes the possibility of loading multiple items before performing a drop; battery constraints take into account the time necessary to charge or change the battery of the AGVs; and conflict-free routing tries to avoid conflicts and deadlocks in a multi-AGV system.

All these problems are not purely academic ones. One of the authors (F. Gallesi) has a direct and daily experience of all such problems as a R&D Engineer of the E80 Group, a company located in Italy that is one of the world-leading companies in the AGVs market. The company deals with large-dimensional AGV systems (up to 200 AGVs in a single plant) in highly dynamic environments, where information changes quickly and, therefore, solutions need to be recomputed every few seconds.

Our contribution is structured as follows. In Section 2.2, we review existing surveys about different approaches for scheduling AGVs. In Section 2.3, we provide a problem definition and a possible formulation of the problem as a PDP. This formulation is the one employed by the E80 Group. Finally, in Section 2.4, we discuss different open problems and research opportunities related to the considered optimization problem.

2.2 Books and surveys on scheduling AGVs

Many surveys addressed the problem of scheduling AGVs in the last twenty years. Lee, Lei, and Pinedo [57], for instance, reviewed scheduling with a 1-job-on-r-machine pattern, which means that one job can be processed simultaneously by r machines; machine scheduling with availability constraints; developments in local search techniques; and practical applications of the scheduling problem. Among these practical applications, they covered AGV scheduling.

Bordelon Hoff and Sarker [14] studied AGV guide paths, including dispatching

rules and related issues such as idle vehicle location and location of pickup and delivery stations. In the same year, Ganesharajah, Hall, and Sriskandarajah [37] summarized some exact and heuristic procedures for flow path design and operational issues, including job scheduling, AGV dispatching, and conflict-free routing. A few years later, Qiu et al. [75] considered scheduling and routing problems with AGVs, providing a review of the main works on the two inter-related problems.

In 2006, Vis [93] proposed a literature review on design and control of AGVs in manufacturing, distribution, transshipment, and transportation systems. Traditional tactical and operational issues were studied in this review, namely flow-path layout, number and location of pickup and delivery points, vehicle requirements, dispatching, routing, and scheduling of vehicles. Furthermore, they also considered vehicle control, conflict avoidance, positioning of idle vehicles, battery and failure management. The design and control of AGVs have been also studied by Le-Anh and De Koster [56], who reviewed and classified key decision models. Some covered topics are guide-path design, scheduling, battery management, and routing.

Some years later, Godinho Filho, Barco, and Tavares Neto [40] reviewed genetic algorithms applied to solve scheduling problems in flexible manufacturing systems (FMS), including scheduling and routing of AGVs. In addition, Fazlollahtabar and Saidi-Mehrabad [33] discussed different methodologies to optimize the scheduling and routing of AGVs, classifying them into exact, heuristic, and metaheuristic approaches. Furthermore, they suggested dedicating more attention to the simultaneous scheduling of different types of material handling equipment in large AGV systems. Kaoud, El-Sharief, and El-Sebaie [51] proposed a review of the studies regarding the scheduling of AGVs in job shop, flow shop, and container terminals applications. After surveying the main works in these three areas, they presented some solutions and directions for future research considering scheduling problems. Similarly, Xie and Allen [95] studied scheduling problems with AGVs in industrial and manufacturing applications, but they focused on the job shop with material handling problems.

In 2020, Schmidt et al. [80] covered the decentralized control strategies for task allocation, empty vehicle balancing, and routing of AGVs. In particular, they focused on vehicle-based in-house transport systems, like AGVs and overhead hoist transport. In the same year, De Ryck, Versteyhe, and Debrouwere [29] elaborated a complete review of the state-of-art regarding all AGV-related control methods, with special attention to decentralized control strategies. They proposed a decomposition for the AGV control into five core tasks, namely task allocation, localization, path planning, motion planning, and vehicle management. For each core task, they provided an extensive review of algorithms and techniques used in the literature.

Very recently, Sun et al. [85] provided definitions of the main research topics in vehicle transportation of the AGV-based automated container terminal (ACT), including equipment scheduling, path planning, exception handling, and vehicle management. A literature review was presented for each topic, and some directions for future research were proposed.

As we can note, scheduling AGVs is a widely studied problem in the literature. Hence, we propose a collection of the more recent and interesting challenges and opportunities on this problem in Section 2.4, providing a guide for future research.

2.3 A mathematical model for AGV scheduling problem

In this section, we focus on the AGV scheduling problem, proposing a possible mathematical model. Although there might be alternative formulations (see, e.g., Bunte and Kliewer [16]), here we model the scheduling problem as a PDP as done by the E80 Group for their real cases.

The PDP considers a directed graph $G = (V, A)$, where $V = F \cup L \cup N$ with: F a set of starting nodes; L a set of possible end nodes, strategic positions to avoid conflicts; N a set of request nodes. Set N is further partitioned into a set P of pickup nodes and a set D of delivery nodes. Let K be the set of requests. With each request $k \in K$ we associate a pickup node $p_k \in P$ and a delivery node $d_k \in D$. We assume that $p_k \neq p_h$, $d_k \neq d_h$ for each $h, k \in K$, $h \neq k$. Moreover, for each $i \in N$, we denote by $k(i)$ the index of the request $k \in K$ associated with node i. A fleet of identical vehicles, represented by set M, is managed to serve the requests, and each vehicle $m \in M$ starts from a specific position $f_m \in F$ (being $|M| = |F|$), serves a number of requests (possibly zero), and ends the route at some $j \in L$, with $|L| \geq |F|$. Hence, we can define the set of arcs as $A = \{(i,j) : i \in F, j \in P\} \cup \{(i,j) : i = p_k, j = d_k\} \cup \{(i,j) : i \in D, j \in P, k(i) \neq k(j)\} \cup \{(i,j) : i \in D, j \in L\} \cup \{(i,j) : i \in F, j \in L\}$. Note that a vehicle traveling from a node in F to a node in L is a vehicle which does not serve any request. We denote by q_i the service time at node $i \in V$. In addition, a deterministic travel time t_{ij} and a traveling distance c_{ij} are associated with each $(i,j) \in A$. Let $[a_k, b_k]$ be the time window for the request $k \in K$, which translates into $[a_{p_k}, b_{p_k}] = [0, b_k - t_{p_k,d_k} - q_{d_k}]$ for p_k and $[a_{d_k}, b_{d_k}] = [a_k, b_k]$ for d_k. The pickup can be arranged at any time. However, please note that if the completion time taken exceeds $b_k - t_{p_k,d_k} - q_{d_k}$, there may be a delay. In case a vehicle arrives at node $i \in N$ before the start time a_i, it waits until a_i to start the service. Otherwise, if a vehicle finishes the service at node $i \in N$ after b_i, we consider the delay R_i as the difference between the completion time and b_i. In this problem, all requests must be served, and the objective function minimizes a suitable trade-off between the total delay and the total distance traveled by the vehicles.

We introduce the following decision variables:

- x_{ij}^m, equal to 1 if arc (i,j) is traversed by vehicle m, 0 otherwise;
- S_i, indicating the starting time of service at node $i \in V$.
- R_i, indicating the delay at node $i \in V$.

The model can be formulated as follows:

$$\min \quad \alpha\Big(\sum_{i\in N} R_i\Big) + \beta\Big(\sum_{m\in M}\sum_{(i,j)\in A} c_{ij}x_{ij}^m\Big) \tag{2.1}$$

$$\text{s.t.} \quad \sum_{\substack{j\in V:\\ (f_m,j)\in A}} x_{f_m,j}^m = 1 \qquad \forall m \in M \tag{2.2}$$

$$\sum_{m\in M}\sum_{i\in V:(i,j)\in A} x_{ij}^m = 1 \qquad \forall j \in P \tag{2.3}$$

$$\sum_{\substack{i\in V:\\ (i,p_k)\in A}} x_{i,p_k}^m = x_{p_k,d_k}^m = \sum_{\substack{j\in V:\\ (d_k,j)\in A}} x_{d_k,j}^m \qquad \forall k \in K, m \in M \tag{2.4}$$

$$\sum_{m\in M}\sum_{\substack{i\in V:\\ (i,j)\in A}} x_{ij}^m \le 1 \qquad \forall j \in L \tag{2.5}$$

$$S_j \ge S_i + \Big(q_i + t_{ij} + H\Big)\sum_{m\in M} x_{ij}^m - H \qquad \forall (i,j) \in A \tag{2.6}$$

$$a_i \le S_i \qquad \forall i \in N \tag{2.7}$$

$$R_i \ge S_i + q_i - b_i \qquad \forall i \in N \tag{2.8}$$

$$x_{ij}^m \in \{0,1\} \qquad \forall (i,j) \in A, m \in M \tag{2.9}$$

$$S_i, R_i \ge 0 \qquad \forall i \in V \tag{2.10}$$

The objective function (2.1) minimizes the weighted sum of the total delay and of the distance traveled by the vehicles. The values α, β are such that $\alpha, \beta \in [0,1]$ and $\alpha + \beta = 1$. They assign a weight to each one of the two components of the objective function. For instance, since the E80 Group prefers to reduce the total delay as much as possible and gives less importance to the distance traveled, a value α close to 1 is employed. Constraints (2.2)–(2.5) define the flow. The route for every vehicle m starts from a fixed node $f_m \in F$, visits a number of requests (the number can also be zero if the vehicle travels along an arc joining a node in F with a node in L), and ends at some node $j \in L$. Constraints (2.2) ensure that each vehicle moves to a pickup node or to an end node. Constraints (2.3) guarantee that a pickup node is served by a single vehicle. Constraints (2.4) enforce that the same vehicle visits the pickup and the delivery node associated with a given request. Constraints (2.5) state that every vehicle ends the route in a different position $j \in L$. The schedule feasibility is guaranteed by constraints (2.6), (2.7) and (2.8). Note that constraints (2.6) also eliminate sub-tours (here H is a big enough constant). Constraints (2.9) and (2.10) define the domain of the variables. As a final remark, we note that we have presented a general PDP formulation with AGVs, but the model, with the addition of a few variables and constraints, can be extended to take into account other attributes, such as multi-load vehicles or battery management.

2.4 Challenges and opportunities in scheduling AGVs

The problem of scheduling AGVs has been extensively studied in the recent literature, as indicated in Section 2.2. Nevertheless, there are research gaps and suggestions for future work. In this section, we provide some open challenges and opportunities for future research on this topic.

2.4.1 Plain scheduling

Scheduling is one of the most fundamental problems related to the management of AGVs. According to the definition introduced by Qiu et al. [75], the problem aims at dispatching a set of AGVs to perform a batch of tasks under some constraints, such as deadlines, priorities, and others. In particular, our focus in this section is on the version of the problem that includes only the main restrictions and does not consider additional attributes (as, e.g., battery management).

He, Aggarwal, and Nof [42] develop a Differentiated Probabilistic Queuing algorithm for assigning orders to AGVs in sequence. Each order has an AGV assignment probability, and a priority. The algorithm has been computationally tested and obtained favorable results. The authors suggest, as future work, relaxing some of the assumptions made for the problem, such as the comparisons between physical and digital storage services cases, and the different combinations of product and service level pricing.

Rahman and Nielsen [76] propose a methodology for scheduling automated transport vehicles based on a mixed integer programming model and a genetic algorithm. They encourage future researchers to consider realistic aspects, namely sequence-dependent setups, buffer size constraints, process interruptions, and shortages in material supplies.

Djenadi and Mendil [31] develop a strategy for task allocation to AGVs based on an Iterated Local Search metaheuristic, considering distributed energy management. The results show that the proposed approach effectively balances productivity maximization and energy consumption minimization. Nervetheless, more complex problems, considering aspects like batteries with non-linear behavior, different vehicle types, and mechanisms of collaborative robotics, are indicated as interesting topics for future research.

2.4.2 Scheduling with battery management

An aspect that is not considered in some works is battery management. The time necessary to charge or to change the battery of the AGVs, as well as the determination of the best locations for the recharging stops, can affect other problems related to the control of AGVs. Therefore, there are several challenges and open questions for future work on this topic.

Boccia et al. [13] study a particular AGV scheduling problem whose objective is to minimize the makespan, satisfying battery constraints. They devise a mathematical model and a matheuristic. As possible extensions, they suggest the inclusion of partial charging operations and the residual energy in the computation of the speed of the AGVs. In addition, they also suggest considering different objective functions besides the makespan minimization.

Singh et al. [81] study a more complex AGV scheduling problem, considering transport requests with soft due dates, different penalty charges, a heterogeneous fleet of AGVs, and partial charging of AGVs with a critical battery threshold. They propose a mathematical model and a matheuristic based on the adaptive large neighborhood search (ALNS). As possible future works, they suggest extending their approach to consider multi-load AGVs and path planning during the scheduling process.

De Ryck et al. [28] present a decentralized task allocation architecture for a fleet of AGVs, based on a sequential single-item auction principle and resource (battery)

management. More specifically, they extend their previous work De Ryck, Versteyhe, and Shariatmadar [30], including a possible extra cost of charging in the bidding mechanism. As a possible topic for future research, they suggest considering routing information and uncertainty in the bidding process.

2.4.3 Scheduling multi-load vehicles

An AGV system may include different kinds of vehicles. In particular, the multi-load vehicles can pick up multiple items before performing a drop. This characteristic increases the complexity of the management of AGVs. Hence, several challenges are still open.

Chawla et al. [20] study the problem of dispatching and scheduling multi-load AGVs in FMS. They put forward a simulation experiment to evaluate the performance of five types of job selection dispatching rules, considering different-sized FMS layouts. In future works, they propose to extend their analysis, including other control aspects, such as the number of loading/unloading points and types of FMS layouts.

Dang et al. [24] devise a hybrid ALNS algorithm to solve a scheduling multi-load AGVs. They consider a heterogeneous fleet and battery constraints as additional attributes, besides the availability of vehicles able to transport multiple loads. For future studies, they suggest to investigate multi-objective criteria and collision-free trajectories.

In the same year, Zou, Pan, and Tasgetiren [99] tackle an AGV scheduling problem considering vehicles with multiple compartments. This problem aims at minimizing the total cost, including travel, service and vehicle costs. To solve the problem, they put forward an iterated greedy algorithm. For future work, they suggest to consider additional attributes for the problem, such as release time, pickup and delivery, and multi-objective, as well as enhancing the proposed algorithm.

Lin et al. [60] develop a task scheduling optimization method for multi-load AGVs systems that aims to minimize the number of AGVs used, travel time, and occurrences of conflicts among the vehicles. The future challenges they propose concern the dynamic rescheduling of the tasks to handle unexpected faults and taking into account dynamic scheduling requirements such as changing throughput and delivery time, in addition to temporary dynamic tasks.

2.4.4 Integrated scheduling and production with AGVs

For most systems, a plain AGV scheduling is not enough to attain efficiency. Indeed, many systems include other automated machines that need to be integrated with the AGVs. This leads to several research questions, as indicated in what follows.

Heger and Voß [44] address the scheduling and dispatching problem with dynamic priority for a flexible job shop scenario, including multi-purpose machines and AGVs. The idea is to use more than one machine to produce the same product and to use the AGVs to transport it. They intend to study dynamic aspects like product mix changes in future works.

In the same year, Lyu et al. [64] focus on scheduling AGVs and machines considering, simultaneously, optimal number of AGVs, shortest transportation time, path planning, and conflict-free routing. They implement a combined approach that includes a genetic algorithm and the Dijkstra algorithm with time windows, and whose goal is finding a balance between the minimal makespan and the number of

AGVs. Furthermore, they intend to consider job sequence and dynamic scheduling problems in future research.

Zhong et al. [98] tackle the scheduling of AGVs considering their impact on ACT, including quay and yard cranes. With the aim of preventing conflicts and deadlocks in the system, they consider an integrated AGV scheduling and path planning problem, which they model as a mixed integer program and then solve by means of a Hybrid Genetic-Particle Swarm Optimization Algorithm. As future research contributions, they suggest finding heuristic algorithms fast enough for solving dynamic real-time scheduling, in some cases supported by artificial intelligence or machine learning.

Finally, Chen et al. [22] focus on the crane and the AGV coordination and scheduling problem in a container terminal and adopt a market-driven Alternating Direction Method of Multipliers approach to solve it. As future research directions, they suggest including other agents in the schedule planning (such as cargo trains, vessels, and forklifts) and use different layouts, time windows, and container stacking sequence constraints.

2.5 Conclusions

In this work, we consider AGV systems and focus on the scheduling problem modeled as a pickup and delivery problem with AGVs. After a study of the literature regarding this topic, we present the mathematical formulation of the problem used in the applications of the E80 Group, which aims to minimize the delay concerning the time windows and the traveled distance of the fleet of vehicles. Finally, we highlight the challenges and opportunities described by the more recent works on the more interesting variants of the AGV scheduling problem. As future research directions, we intend to extend our analysis to include other variants of the problems together with the related literature.

Acknowledgements

The authors thank E80 Group SpA for financial support and for sharing relevant information that helped the development of this research. The third author also gratefully acknowledges financial support under the National Recovery and Resilience Plan (NRRP), Mission 04 Component 2 Investment 1.5–NextGenerationEU, Call n. 3277, Award n. 0001052. The last author was supported by the Paraíba State Research Foundation (FAPESQ), grant 261/2020, by the Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), grants 406245/2021-5 and 309580/2021-8.

Chapter 3

Recent advances on Pickup and Delivery Problems with AGVs*

In this chapter, we extend the Chapter 2 considering the coordination of a fleet of AGVs in general. After a review of books and surveys that constitute the current state of the art in AGV systems, we propose two variants of the model proposed in the previous chapter: PDP with battery managemnt and PDP with multi-load vehicles. Finally, we present a list of papers that highlight current challenges and indicate future research directions found in the literature.

3.1 Introduction

In the last decades, AGVs have become a common equipment to transport materials, goods and containers in logistic systems, changing the work habits of many companies. The AGVs provide efficient and flexible solutions for transportation and manufacturing systems, and their management has consequently become a crucial logistic activity. These robots navigate through dynamic environments with a precision and adaptability that outperforms traditional manual methods. The usage of AGVs brings many benefits for companies, considering the increase in productivity and cost-effectiveness to a reduction in errors and delays. Their versatility allows them to integrate into diverse industrial settings, adapting to varying demands and operational requirements. This adaptability is particularly crucial in the contemporary context, where technologies change fast and the need for agile logistics has become a defining characteristic of competitive enterprises. The interest towards AGVs and the optimization problems arising from their use is testified by the rapid increase in the number of works devoted to these topics. The rapid proliferation of publications focused on AGVs underscores a collective recognition within the academic and industrial communities of the impact these autonomous vehicles have on logistics and manufacturing systems.

In particular, as defined by De Ryck, Versteyhe, and Debrouwere [29], there are five main problems related to AGVs: task allocation/scheduling, localization, path planning, motion planning, and vehicle management for AGV coordination. More precisely, task allocation/scheduling considers assigning a set of tasks to the AGVs; localization refers to the position of each AGV in the environment; path planning defines the shortest path to the destination; motion planning avoids collisions with other AGVs and static objects; and vehicle management considers the status of the AGVs, focusing on the errors and the maintenance. Some variants of these problems

*The results of this chapter appears in the technical report: Gallesi, F., Praxedes, R., Iori, M., Locatelli, M., and Subramanian, A. "Recent advances on Pickup and Delivery Problems with AGVs".

consider additional attributes and environmental aspects, such as multi-load, battery constraints, and conflict-free routing, just to mention a few. Multi-load scheduling includes the possibility of loading multiple items before performing a drop; battery constraints take into account the time necessary to charge or change the battery of the AGVs; and conflict-free routing tries to avoid conflicts and deadlocks in a multi-AGV system.

All these problems are not purely academic ones. One of the authors (F. Gallesi) has a direct and daily experience of all such problems as a R&D Engineer of the E80 Group, a company located in Italy that is one of the world-leading companies in the AGVs market. The E80 Group specializes in the development of automated and integrated intralogistics solutions for manufacturers of consumer goods, with a particular interest in the coordination of AGVs in automated plants. The company deals with large-dimensional AGV systems (up to 200 AGVs in a single plant) in highly dynamic environments, where information changes quickly and, therefore, solutions need to be recomputed every few seconds. The fleet of AGVs serves the transport from the raw material and the processing area to the packaging robots, and moves the finished product to the stocking and the shipping area. The company wants to be close to worldwide customers and for this reason, it has opened 14 branches and subsidiaries in the world.

Our contribution is structured as follows. In Section 3.2, we review existing surveys about different approaches for design and control of AGVs. In Section 3.3, we provide a problem definition and a possible formulation of scheduling AGVs as a PDP. This formulation is the one employed by the E80 Group. Finally, in Section 3.4, we discuss different open problems and research opportunities related to this challenging research field.

3.2 Books and surveys on design and control of AGVs

Many surveys addressed the optimization problems related to the design and control of AGVs in the last twenty years. Co and Tanchoco [23], for instance, started focusing on the AGVs vehicle management problem. Moreover, King and Wilson [52] divided the literature regarding AGVs into three categories: system design, routing and scheduling, and justification and implementation. The first corresponds to the papers that examine layout and fleet issues, the second covers the methods for routing and scheduling, and the third discusses some issues related to the reasons for using AGVs and how to implement them. A few years later, Lee, Lei, and Pinedo [57] reviewed scheduling with a 1-job-on-r-machine pattern, which means that one job can be processed simultaneously by r machines; machine scheduling with availability constraints; developments in local search techniques; and practical applications of the scheduling problem. Among these practical applications, they covered AGV scheduling.

Bordelon Hoff and Sarker [14] studied AGV guide paths, including dispatching rules and related issues such as idle vehicle location and location of pickup and delivery stations. In the same year, Ganesharajah, Hall, and Sriskandarajah [37] summarized some exact and heuristic procedures for flow path design and operational issues, including job scheduling, AGV dispatching, and conflict-free routing. A few years later, Qiu et al. [75] considered scheduling and routing problems with AGVs, providing a review of the main works on the two inter-related problems. On the other hand, papers regarding facility planning and material handling decisions in the context of loop-based material flow systems were reviewed by Asef-Vaziri and

Laporte [5]. Among these decisions, issues related to fleet sizing and operating of AGVs, namely the home location of idle vehicles, blocking and collision avoidance, multi-load, and others, were also covered.

In 2006, Vis [93] proposed a literature review on design and control of AGVs in manufacturing, distribution, transshipment, and transportation systems. Traditional tactical and operational issues were studied in this review, namely flow-path layout, number and location of pickup and delivery points, vehicle requirements, dispatching, routing, and scheduling of vehicles. Furthermore, they also considered vehicle control, conflict avoidance, positioning of idle vehicles, battery and failure management. The design and control of AGVs have also been studied by Le-Anh and De Koster [56], who reviewed and classified key decision models. Some covered topics are guide-path design, scheduling, battery management, and routing. Similarly, Ali and Khan [1] presented implementation issues (number of vehicles and the location of pickup and drop-off points, types of AGVs, path generation, collision and deadlock avoidance) related to AGVs in Flexible Manufacturing Systems (FMS). Moreover, they also tackled the integration between AGVs and FMS. On the other hand, Stahlbock and Voß [83] provided a comprehensive survey on routing problems in the Container Terminalss (CTs) context, in which AGVs have a fundamental role. The authors explained some general concerns about Vehicle Routing Problems (VRP) and the structures of CTs. They further described various VRPs related to CT operations, dedicating one section only for horizontal transport with AGVs.

Some years later, Godinho Filho, Barco, and Tavares Neto [40] reviewed genetic algorithms applied to solve scheduling problems in FMS, including scheduling and routing of AGVs. In addition, Fazlollahtabar and Saidi-Mehrabad [33] discussed different methodologies to optimize the scheduling and routing of AGVs, classifying them into exact, heuristic, and metaheuristic approaches. Furthermore, they suggested dedicating more attention to the simultaneous scheduling of different types of material handling equipment in large AGV systems. In the same year, Xie and Allen [95] studied scheduling problems with AGVs in industrial and manufacturing applications, focusing on the job shop with material handling problems.

In 2020, Schmidt et al. [80] covered the decentralized control strategies for task allocation, empty vehicle balancing, and routing of AGVs. In particular, they focused on vehicle-based in-house transport systems, like AGVs and overhead hoist transport. In the same year, De Ryck, Versteyhe, and Debrouwere [29] elaborated a complete review of the state-of-art regarding all AGV-related control methods, with special attention to decentralized control strategies. They proposed a decomposition for the AGV control into five core tasks, namely task allocation, localization, path planning, motion planning, and vehicle management. For each core task, they provided an extensive review of algorithms and techniques used in the literature.

One year after, Rashidi, Matinfar, and Parand [77] proposed a review of applications, problem modeling and solutions considering AGVs in container handling and FMS. Moreover, Ivanov et al. [48] performed a multidisciplinary analysis associating Industry 4.0 with the management of AGVs. Fragapane et al. [35] covered the decentralized decision-making process for planning and control of Autonomous Mobile Robots (AMRs). In addition to the previous works, Lu et al. [63] provided a literature review regarding multi-AGVs systems management. They divided those systems into three problems: scheduling, dispatching, and routing. For each of them, they highlighted several works according to the resource-oriented, problem-oriented, and goal-oriented issues that appear in multi-AGV systems.

Very recently, three new surveys appeared in the literature. Naeem, Gheith, and Eltawil [71] covered the integrated scheduling of different kinds of equipment in

Automated Container Terminals (ACT), which is divided into three main problems: integrated scheduling of quay cranes and AGVs, yard cranes and AGVs, and all of them together. For all of these problems, a comprehensive literature review is provided. In addition, research gaps and future research directions are pointed out. Sun et al. [85] provided definitions of the most important research topics in vehicle transportation of the AGV-based automated container terminal, including equipment scheduling, path planning, exception handling, and vehicle management. A literature review was presented for each topic, and some directions for future research were proposed. Zhang, Chen, and Guo [97] reviewed the application of AGVs in warehouse systems. Among the covered topics, there is a comparison between centralized and decentralized control strategies, warehouse layouts, scheduling and routing AGVs problems, and artificial intelligence applications in this kind of system. Besides, a method combining reinforcement learning and Dijkstra's algorithm was proposed. Table 3.1 summarizes all the books and surveys covered in this section, emphasizing their main covered topics and applications areas on the second and third columns, respectively. It is important to mention that the symbol "——" is used when the problem on the corresponding reference is considered in general way, without specific applications.

3.3 A mathematical model for AGV scheduling problem

In this section, we focus on the AGV scheduling problem, proposing a possible mathematical model. Although there might be alternative formulations (see, e.g., Bunte and Kliewer [16]), here we model the scheduling problem as a PDP as done by the E80 Group for their real cases.

The PDP considers a directed graph $G = (V, A)$, where $V = V_f \cup V_l \cup V_n$ with: V_f a set of starting nodes; V_l a set of possible end nodes, strategic positions to avoid conflicts; V_n a set of request nodes. Set V_n is further partitioned into a set P of pickup nodes and a set D of delivery nodes. Let K be the set of requests. With each request $k \in K$ we associate a pickup node $p_k \in P$ and a delivery node $d_k \in D$. We assume that $p_k \neq p_h$, $d_k \neq d_h$ for each $h, k \in K$, $h \neq k$. Moreover, for each $i \in V_n$, we denote by $k(i)$ the index of the request $k \in K$ associated with node i. A fleet of identical vehicles, represented by set M, is managed to serve the requests, and each vehicle $m \in M$ starts from a specific position $f_m \in V_f$ (being $|M| = |V_f|$), serves a number of requests (possibly zero), and ends the route at some $j \in V_l$, with $|V_l| \geq |V_f|$. Hence, we can define the set of arcs as $A = \{(i,j) : i \in V_f, j \in P\} \cup \{(i,j) : i = p_k, j = d_k\} \cup \{(i,j) : i \in D, j \in P, k(i) \neq k(j)\} \cup \{(i,j) : i \in D, j \in L\} \cup \{(i,j) : i \in V_f, j \in V_l\}$. Note that a vehicle traveling from a node in V_f to a node in V_l is a vehicle which does not serve any request. We denote by q_i the service time at node $i \in V$. In addition, a deterministic travel time t_{ij} and a traveling distance c_{ij} are associated with each $(i,j) \in A$. Let $[a_k, b_k]$ be the time window for the request $k \in K$, which translates into $[a_{p_k}, b_{p_k}] = [0, b_k - t_{p_k,d_k} - q_{d_k}]$ for p_k and $[a_{d_k}, b_{d_k}] = [a_k, b_k]$ for d_k. The pickup can be arranged at any time. However, note that if the completion time taken exceeds $b_k - t_{p_k,d_k} - q_{d_k}$, there might be a delay. In case a vehicle arrives at node $i \in N$ before the start time a_i, it waits until a_i to start the service. Otherwise, if a vehicle finishes the service at node $i \in V_n$ after b_i, we consider the delay R_i as the difference between the completion time and b_i. In this problem, all requests must be served, and the objective function minimizes a suitable trade-off between the total delay and the total distance traveled by the vehicles.

We introduce the following decision variables:

Table 3.1: Books and surveys on pickup and delivery problems with AGVs

Survey/book	Covered topics	Applications
Co and Tanchoco [23] Engineering Costs and Production Economics	Vehicle management problem	——
King and Wilson [52] Production Planning & Control	System design, routing and scheduling, and justification and implementation	——
Lee, Lei, and Pinedo [57] Annals of Operations Research	Scheduling of machines, including AGVs	Material handling transporters
Bordelon Hoff and Sarker [14] Integrated Manufacturing Systems	Guide paths, dispatching rules, idle vehicle location and stations location	——
Ganesharajah, Hall, and Sriskandarajah [37] Annals of Operations Research	Flow path design and operational issues, including AGV dispatching	Manufacturing systems
Qiu et al. [75] International Journal of Production Research	Scheduling and routing problems with AGVs	——
Asef-Vaziri and Laporte [5] European Journal of Operational Research	Facility planning and material handling decisions	Loop-based material flow systems
Vis [93] European Journal of Operational Research	Issues related to the design and control of AGVs	Manufacturing, transshipment distribution, and transportation
Le-Anh and De Koster [56] European Journal of Operational Research	Issues related to the design and control of AGVs	——
Stahlbock and Voß [83] Springer US book	Routing problems (Horizontal transport with AGVs)	Container terminals
Ali and Khan [1] Global Journal of Flexible Systems Management	Issues related to the implementation of AGVs	Flexible manufacturing systems
Godinho Filho, Barco, and Tavares Neto [40] Flexible Services and Manufacturing Journal	Genetic algorithms to solve scheduling problems	Flexible manufacturing systems
Fazlollahtabar and Saidi-Mehrabad [33] Journal of Intelligent & Robotic Systems	Scheduling and routing problems with AGVs	Manufacturing, transshipment distribution, and transportation
Xie and Allen [95] The International Journal of Advanced Manufacturing Technology	Scheduling problem with AGVs	Job shop with material handling
Schmidt et al. [80] Logistics Research	Decentralized control strategies	Automated in-house logistics vehicle systems
De Ryck, Versteyhe, and Debrouwere [29] Journal of Manufacturing Systems	Decentralized control strategies	——
Rashidi, Matinfar, and Parand [77] International Journal of Transportation Engineering	Problem modeling, solution and applications with AGVs	Containers and flexible manufacturing systems
Ivanov et al. [48] International Journal of Production Research	Industry 4.0, including AGVs	——
Fragapane et al. [35] European Journal of Operational Research	Decentralized planning and control of AMRs	——
Lu et al. [63] Computer Modeling in Engineering & Sciences	Scheduling, dispatching, and routing problem with AGVs	——
Naeem, Gheith, and Eltawil [71] Computers & Industrial Engineering	Integrated scheduling of AGVs, quay cranes, and yard cranes	Automated container terminals
Sun et al. [85] IEEE Transactions on Intelligent Transportation Systems	Scheduling, path planning, exception handling, and vehicle management	Automated container terminals
Zhang, Chen, and Guo [97] Computer Modeling in Engineering & Sciences	Issues on AGV applications	Warehouse systems

- x_{ij}^m, equal to 1 if arc (i,j) is traversed by vehicle m, 0 otherwise;
- S_i, indicating the starting time of service at node $i \in V$.
- R_i, indicating the delay at node $i \in V$.

The model can be formulated as follows:

$$\min \quad \alpha\Big(\sum_{i\in V_n} R_i\Big) + \beta\Big(\sum_{m\in M}\sum_{(i,j)\in A} c_{ij}x_{ij}^m\Big) \tag{3.1}$$

$$\text{s.t.} \quad \sum_{\substack{j\in V:\\ (f_m,j)\in A}} x_{f_m,j}^m = 1 \qquad m \in M \tag{3.2}$$

$$\sum_{m\in M}\sum_{i\in V:(i,j)\in A} x_{ij}^m = 1 \qquad j \in P \tag{3.3}$$

$$\sum_{\substack{i\in V:\\ (i,p_k)\in A}} x_{i,p_k}^m = x_{p_k,d_k}^m = \sum_{\substack{j\in V:\\ (d_k,j)\in A}} x_{d_k,j}^m \qquad k \in K, m \in M \tag{3.4}$$

$$\sum_{m\in M}\sum_{\substack{i\in V:\\ (i,j)\in A}} x_{ij}^m \le 1 \qquad j \in V_l \tag{3.5}$$

$$S_j \ge S_i + \big(q_i + t_{ij} + H\big)\sum_{m\in M} x_{ij}^m - H \qquad (i,j) \in A \tag{3.6}$$

$$a_i \le S_i \qquad i \in V_n \tag{3.7}$$

$$R_i \ge S_i + q_i - b_i \qquad i \in V_n \tag{3.8}$$

$$x_{ij}^m \in \{0,1\} \qquad (i,j) \in A, m \in M \tag{3.9}$$

$$S_i, R_i \ge 0 \qquad i \in V \tag{3.10}$$

Objective function (3.1) minimizes the weighted sum of the total delay and of the distance traveled by the vehicles. The values α, β are such that $\alpha, \beta \in [0,1]$ and $\alpha + \beta = 1$. They assign a weight to each one of the two components of the objective function. For instance, since the E80 Group prefers to reduce the total delay as much as possible and gives less importance to the distance traveled, a value α close to 1 is employed. Constraints (3.2)–(3.5) define the flow. The route for every vehicle m starts from a fixed node $f_m \in V_f$, visits a number of requests (the number can also be zero if the vehicle travels along an arc joining a node in V_f with a node in V_l), and ends at some node $j \in V_l$. Constraints (3.2) ensure that each vehicle moves to a pickup node or to an end node. Constraints (3.3) guarantee that a pickup node is served by a single vehicle. Constraints (3.4) enforce that the same vehicle visits the pickup and the delivery node associated with a given request. Constraints (3.5) state that every vehicle ends the route in a different position $j \in V_l$. The schedule feasibility is guaranteed by constraints (3.6), (3.7) and (3.8). Note that constraints (3.6) also eliminate sub-tours (here H is a big enough constant). Constraints (3.9) and (3.10) define the domain of the variables. Figure 3.1 shows an example of a solution for the PDP as described above. Two AGVs are in a warehouse and four transport requests are available. The solution proposes a route for the vehicles. The depicted routes in the figure consist of a sequence of requests, each encompassing a pickup and a corresponding delivery point, and then concluding with a home position L. The AGV1 visits $P1$, $D1$, $P2$, $D2$, L. The AGV2 visits $P3$, $D3$, $P4$, $D4$, L. As a final remark, we note that we have presented a general PDP formulation with AGVs, but the model, with the addition of a few variables and constraints, can

be extended to take into account other attributes, such as battery management or multi-load vehicles.

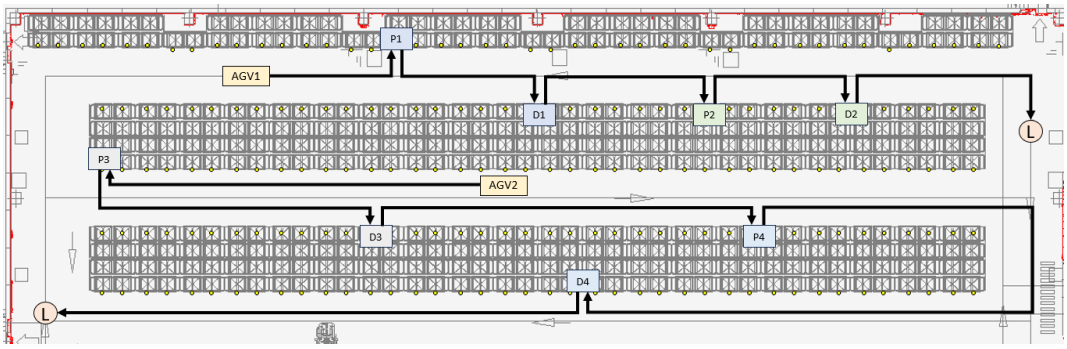


Figure 3.1: Example of pickup and delivery solution.

3.3.1 Model extension

The extension of the model for battery management requires the incorporation of a new set of nodes, denoted as V_b, which represent the facilities housing the battery chargers, and of an additional set of arcs $A' = \{(i,j) : i \in V_f \cup D, j \in V_b\} \cup \{(i,j) : i \in V_b, j \in P \cup V_l\}$. Consequently, the augmented set of available arcs becomes $A \cup A'$. We then set the maximum charging level B of a battery, and the variable b_{ij}^m defining the residual battery level of vehicle $m \in M$ after traversing the arc $(i,j) \in A \cup A'$. Upon entering node $i \in V_b$, a vehicle recharges its battery level to the maximum, B. It is assumed that each vehicle $m \in M$ begins its journey from f_m with a battery level sufficient to reach at least one battery charger. Furthermore, we consider the consumption of the battery as linear function of the travel time t_{ij} on the arc (i,j). The additional constraints necessary to extend the model for battery management are as follows:

$$\sum_{i \in V\,:\,(i,j) \in A'} x_{ij}^m = \sum_{h \in V\,:\,(j,h) \in A'} x_{jh}^m \qquad j \in V_b, m \in M \tag{3.11}$$

$$b_{ij}^m + x_{ij}^m t_{ij} \leq \sum_{h \in V\,:\,(h,i) \in A \cup A'} b_{hi}^m \qquad (i,j) \in A \cup A', m \in M \tag{3.12}$$

$$b_{ij}^m = (B - t_{ij}) x_{ij}^m \qquad i \in V_b, (i,j) \in A', m \in M \tag{3.13}$$

$$0 \leq b_{ij}^m \leq B x_{ij}^m \qquad (i,j) \in A \cup A', m \in M. \tag{3.14}$$

Constraints (3.11) express the inflow and outflow of vehicles at the battery charger facility, emphasizing that there is no limitation on the number of times a vehicle can utilize a battery charger. Constraints (3.12) and (3.13) ensure the feasibility of the schedule. Constraints (3.14) define the domain of the variables.

In the multi-load version, where a vehicle has the capability to pickup more than one task simultaneously, one has to adapt constraints (3.4) to allow visits to other nodes between the pickup and delivery of a request. First, we need to update the set of arcs in the model by considering $A'' = \{(i,j) : i \in V_n, j \in V_n, i \neq j\}$, with the resulting set of arcs denoted as $A \cup A''$. Constraints (3.4) can then be modified as follows:

$$\sum_{\substack{i \in V: \\ (i,p_k) \in A \cup A''}} x_{i,p_k}^m = \sum_{\substack{j \in V: \\ (d_k,j) \in A \cup A''}} x_{d_k,j}^m \qquad k \in K, m \in M. \tag{3.15}$$

In view of this, we can introduce the maximum capacity of a vehicle, denoted as L, and the variable l_i^m defining the residual capacity of vehicle $m \in M$ at node $i \in V$. Additionally, we can define u_i as the capacity required by node $i \in V$. The capacity is assumed to be negative for pickup nodes, positive for delivery nodes, and null for the remaining ones. The following constraints are necessary to extend the model to accommodate multi-load vehicles:

$$\sum_{i \in V:(i,j) \in A \cup A''} x_{ij}^m = \sum_{h \in V:(j,h) \in A \cup A''} x_{jh}^m \qquad j \in V_n, m \in M \tag{3.16}$$

$$S_{p_k} \leq S_{d_k} \qquad k \in K \tag{3.17}$$

$$l_j^m \leq l_i^m + u_i x_{ij}^m \qquad (i,j) \in A \cup A'', m \in M \tag{3.18}$$

$$0 \leq l_i^m \leq L \sum_{j \in V:(j,i) \in A \cup A''} x_{ji}^m \qquad i \in V, m \in M. \tag{3.19}$$

Constraints (3.16) and (3.17) are used to delineate the flow and guarantee that each delivery node is not visited before the corresponding pickup node, respectively. Constraints (3.18) articulate the residual capacity at each node, while constraints (3.19) establish the domain for the associated variable. Of course, the two variants of the basic PDP model detailed in this section are not the only the possible ones. Further variants could be considered, like, e.g., those taking into account possible conflicts between AGVs at nodes and arcs of the graph.

3.4 Challenges and opportunities in design and control of AGVs

The design and control of AGVs have been extensively studied in the recent literature, as discussed in Section 3.2. Nevertheless, there are research gaps and suggestions for future work. In this section, we provide some open challenges and opportunities for future research based on the most recent references regarding this topic.

3.4.1 Scheduling

Scheduling is one of the most fundamental problems related to the management of AGVs. According to the definition introduced by Qiu et al. [75], the problem aims at dispatching a set of AGVs to perform a batch of tasks under some constraints, such as deadlines, priorities, and others. In this section, we split the problem into plain scheduling, scheduling with battery constraints, and scheduling multi-load vehicles, highlighting the most recent research gaps indicated in the literature. For readers interested in more details about the scheduling of AGVs, we suggest referring to the surveys of Fazlollahtabar and Saidi-Mehrabad [33], Lu et al. [63], Qiu et al. [75], Sun et al. [85], and Xie and Allen [95].

Plain scheduling

He, Aggarwal, and Nof [42] developed a Differentiated Probabilistic Queuing algorithm for assigning orders to AGVs in sequence. Each order has an AGV assignment probability, and a priority. The algorithm has been computationally tested and obtained favorable results. The authors suggest, as future work, relaxing some of the assumptions made for the problem, such as the comparisons between physical and

digital storage services cases, and the different combinations of product and service level pricing.

A methodology for scheduling automated transport vehicles based on a mixed integer programming model and a genetic algorithm was proposed by Rahman and Nielsen [76]. They encourage future researchers to consider realistic aspects, namely sequence-dependent setups, buffer size constraints, process interruptions, and shortages in material supplies.

Recently, Djenadi and Mendil [31] developed a strategy for task allocation to AGVs based on an Iterated Local Search metaheuristic, considering distributed energy management. The results show that the proposed approach effectively balances productivity maximization and energy consumption minimization. Nevertheless, more complex problems, considering aspects like batteries with non-linear behavior, different vehicle types, and mechanisms of collaborative robotics, are indicated as interesting topics for future research.

Scheduling with battery constraints

De Ryck et al. [28] presented a decentralized task allocation architecture for a fleet of AGVs, based on a sequential single-item auction principle and resource (battery) management. More specifically, they extended their previous work De Ryck, Versteyhe, and Shariatmadar [30], including a possible extra cost of charging in the bidding mechanism. As a possible topic for future research, they suggest considering routing information and uncertainty in the bidding process.

Moreover, Boccia et al. [13] studied a particular AGV scheduling problem whose objective is to minimize the makespan, satisfying battery constraints. They devised a mathematical model and a matheuristic. As possible extensions, they suggest the inclusion of partial charging operations and the residual energy in the computation of the speed of the AGVs. In addition, they also suggest considering different objective functions besides the makespan minimization.

A more complex AGV scheduling problem was studied by Singh et al. [81], considering transport requests with soft due dates, different penalty charges, a heterogeneous fleet of AGVs, and partial charging of AGVs with a critical battery threshold. They proposed a mathematical model and a matheuristic based on the Adaptive Large Neighborhood Search (ALNS). As possible future works, they suggest extending their approach to consider multi-load AGVs and path planning during the scheduling process.

Chen, Chen, and Teng [21] proposed a two-stage simulation-optimization approach for AGV systems with charging mechanisms. In the first stage, they aim to determine the number of required AGVs, considering fixed guided-path constraints. The second stage optimizes some operational issues, such as the charging system, dispatching, positioning, and routing rules. As future research direction, one possibility can be disregarding some assumptions in the simulation model, such as the characteristics of the products, the capacity and setup times of the machines, and the fixed guided-path constraints.

Scheduling multi-load vehicles

Chawla et al. [20] studied the problem of dispatching and scheduling multi-load AGVs in FMS. They put forward a simulation experiment to evaluate the performance of five types of job selection dispatching rules, considering different-sized FMS layouts. In future works, they propose to extend their analysis, including other

control aspects, such as the number of loading/unloading points and types of FMS layouts.

Furthermore, Dang et al. [25] devised a hybrid ALNS algorithm to solve a scheduling multi-load AGVs. They consider a heterogeneous fleet and battery constraints as additional attributes, besides the availability of vehicles able to transport multiple loads. For future studies, they suggest to investigate multi-objective criteria and collision-free trajectories.

Zou, Pan, and Tasgetiren [99] tackled an AGV scheduling problem considering vehicles with multiple compartments. This problem aims at minimizing the total cost, including travel, service and vehicle costs. To solve the problem, they put forward an iterated greedy algorithm. For future work, they suggest to consider additional attributes for the problem, such as release time, pickup and delivery, and multi-objective, as well as enhancing the proposed algorithm.

Recently, Lin et al. [60] developed a task scheduling optimization method for multi-load AGVs systems that aims to minimize the number of AGVs used, travel time, and occurrences of conflicts among the vehicles. The future challenges they propose concern the dynamic rescheduling of the tasks to handle unexpected faults and taking into account dynamic scheduling requirements such as changing throughput and delivery time, in addition to temporary dynamic tasks.

3.4.2 Path planning and conflict avoidance

Once the scheduling decisions are defined, the second step on the AGVs management is determining the path to be traveled by them. Moreover, path planning methods should consider the possibility of conflicts and deadlocks when multiple vehicles are in the system, as suggested by De Ryck, Versteyhe, and Debrouwere [29] and Schmidt et al. [80]. In what follows, we provide several promising avenues of research concerning path planning and conflict avoidance in the context of AGVs.

Kabir and Suzuki [50] presented a comparative study evaluating four heuristics designed for routing AGVs to battery stations, as an extension of their previous work Kabir and Suzuki [49]. The objective of this study is to demonstrate methods for enhancing productivity by minimizing both travel distances and waiting times at battery stations. The authors have considered different heuristic criteria by selecting the: (i) nearest battery station; (ii) minimum delay battery station; (iii) nearest battery station from the initial point; and (iv) nearest battery station to the delivery point. This comparative analysis offers insights into the efficiency of each heuristic in managing AGV battery operations. As a next step, the authors recommend integrating the concept of opportunity swapping and explore the application of different types of facilities, such as warehouses and container terminals.

A dynamic path planning approach to grid-based systems that aims to find the shortest-time paths for AGVs is proposed by Fransen et al. [36]. They suggest to apply the proposed method to grid-based system layouts with bidirectional path segments and non-grid-based systems, as well as an extension of this approach to multi-load AGV systems for sequenced or unsequenced planning as possible future research topics. Another possible topic is related to the vertex weights updating process using future information, instead of only those from the past.

Furthermore, Tang et al. [86] proposed a path planning method for AGVs based on the geometric A-Star algorithm in a port environment, aiming to find the minimum cost collision-free and smooth path. As extensions for this research, they suggest taking into account the size of the AGV; dynamic scenes; the realistic motion

of the AGVs, not only in a main straight line; and implementations of the proposed method in a real-world port environment.

Lastly, a conflict-free multi-load AGVs scheduling problem was studied by Hu, Yang, and Huang [45], who proposed a method based on adjacency combination and the timetables of reservations for solving it. For future works, they suggest studies regarding the influence of the material characteristics on the proposed method, scheduling of heterogeneous multi-load AGVs based on unknown tasks system, and different network topologies of the material transportation systems.

3.4.3 Integrated production with AGVs

For most systems, a plain AGV scheduling is not enough to attain efficiency. Indeed, many systems include other automated machines that need to be integrated with the AGVs. One example of this integration can be found on the work by Naeem, Gheith, and Eltawil [71], where they highlight the current literature about the integration of AGVs with quay and yard cranes. However, various integration scenarios exist beyond this one, leading to several research questions, as discussed in the following.

Heger and Voß [44] addressed the scheduling and dispatching problem with dynamic priority for a flexible job shop scenario, including multi-purpose machines and AGVs. The idea is to use more than one machine to produce the same product and to use the AGVs to transport it. They intend to study dynamic aspects like product mix changes in future works. Similarly, Lyu et al. [64] focused on scheduling AGVs and machines considering, simultaneously, optimal number of AGVs, shortest transportation time, path planning, and conflict-free routing. They implemented a combined approach that includes a genetic algorithm and the Dijkstra algorithm with time windows, and whose goal is finding a balance between the minimal makespan and the number of AGVs. Furthermore, they intend to consider job sequence and dynamic scheduling problems in future research.

The scheduling AGVs with quay and yard cranes on ACT systems was studied by Zhong et al. [98]. With the aim of preventing conflicts and deadlocks in the system, they considered an integrated AGV scheduling and path planning problem, which they model as a mixed integer program and then solve by means of a Hybrid Genetic-Particle Swarm Optimization Algorithm. As future research contributions, they suggest finding heuristic algorithms fast enough for solving dynamic real-time scheduling, in some cases supported by artificial intelligence or machine learning.

Additionally, Chen et al. [22] focused on the crane and the AGV coordination and scheduling problem in a container terminal and adopt a market-driven Alternating Direction Method of Multipliers approach to solve it. As future research directions, they suggest including other agents in the schedule planning (such as cargo trains, vessels, and forklifts) and use different layouts, time windows, and container stacking sequence constraints.

Lastly, Manafi, Tavakkoli-Moghaddam, and Mahmoodjanloo [66] addressed the job-shop scheduling and AGV conflict-free routing problem. They proposed a Centroid Opposition-based Coral Reefs Optimization (COCRO) algorithm to solve the scheduling and routing problems simultaneously. The aim of this integrated problem is to minimize the makespan of all products. Future research might consider different types of AGVs, or a dynamic system able to respond to events not predictable, such as cancellations of tasks or breakdowns.

3.4.4 Miscellaneous

This section is dedicated to the papers that cover other important topics related to the AGV management. We can start highlighting the simulation study conducted by Kabir and Suzuki [49], in which they explore the impact of varying the duration of battery charging for AGVs in order to enhance the flexibility of a manufacturing system. The primary objective is to demonstrate that short-term increases in the production output of a manufacturing system can be achieved by selectively reducing the targeted state of charge for AGV batteries, without the need to augment the overall number of AGVs within the system. Despite the promising nature of their proposed strategy, the authors recognize five limitations in their study, which are interesting topics for future research. These include the omission of AGV breakdown effects, the potential for refining the dispatching policy, the unidirectional assumption in guide-paths, the scope for exploring different battery types (e.g., lithium-ion and nickel-cadmium), and the possibility of employing alternative simulation strategies.

In the scope of muti-load AGV systems, Yan, Jackson, and Dunnett [96] studied the advantages of using this kind of AGV. They developed an advanced form to simulate the operation of the AGV system in various scenarios. The results of this paper suggest future studies on how to increase the capacity of the multi-load vehicles without decreases the performance. For the authors, the multi-load AGV increases the performance of the system when there are flexible loading and unloading points.

Lastly, Ma, Zhou, and Stephen [65] proposed a simulation approach for the configuration of charging stations and battery-powered AGVs. More precisely, they presented two types of layout designs and two types of recharging policies in a port system environment, which is simulated as a discrete event simulation model. Moreover, several computational experiments were conducted to find which strategy (layout and police) is the best. The authors highlighted that this work contributes to the academic and industrial communities, as both strategies can be easily implemented. In future work, they suggest considering AGV scheduling algorithms in their simulation approach.

3.5 Conclusions

In conclusion, the integration of AGVs has revolutionized logistics and manufacturing systems, remodeling the operational context of companies worldwide. The precision and adaptability demonstrated by them in navigating dynamic environments overcome traditional manual methods, providing more efficiency and flexibility. The benefits of AGV deployment, including heightened productivity, cost-effectiveness, and a reduction in errors and delays are fundamental in modern industrial settings. Motivated by the growing number of publications in the scientific world and from an industrial point of view reported by one of the authors, this work highlights the recent advances on design and control of AGVs. After a study of the literature regarding this topic, we present the mathematical formulation of one of the main problems in this field: scheduling AGVs. The model we present is based on PDP formulations, justified by the methodology adopted by the E80 Group in their practical applications. The model is also extended presenting variants for battery management and multi-load vehicles. In these applications, the company aims to minimize the delay concerning the time windows and the traveled distance of the

fleet of vehicles. Besides the proposed model, we highlight the challenges and opportunities described by the more recent works on the more interesting optimization problems in AGV systems.

Chapter 4

A Branch-and-Regret Algorithm for the Same-Day Delivery Problem*

In this chapter, we study a dynamic vehicle routing problem where stochastic customers request urgent deliveries characterized by restricted time windows. The aim is to use a fleet of vehicles to maximize the number of served requests and minimize the traveled distance. The problem is known in the literature as the same-day delivery problem, and it is of high importance because it models a number of real-world applications, including the delivery of online purchases. We solve the same-day delivery problem by proposing a novel branch-and-regret algorithm in which sampled scenarios are used to anticipate future events and an adaptive large neighborhood search is iteratively invoked to optimize routing plans. The branch-and-regret is equipped with four innovation elements: a new way to model the subproblem, a new policy to generate scenarios, new consensus functions, and a new branching scheme. Extensive computational experiments on a large variety of instances prove the outstanding performance of the branch-and-regret, also in comparison with recent literature, in terms of served requests, traveled distance, and computational effort.

4.1 Introduction

In recent years, online purchases have become more and more a common practice to request services or goods at home, changing the habit in which many traditional markets operate. A huge number of e-commerce sellers appeared on the web, and many companies changed their focus to direct-to-consumer deliveries to expand their business. Dayarian, Savelsbergh, and Clarke [27] mentioned that online shopping followed by home delivery has annually increased by around 8.5% in mature markets (e.g., the United States) and close to 300% in developing markets (e.g., India). This trend has been increased by the Covid-19 pandemic, which made people reluctant to leave their homes.

As a consequence, the delivery of online purchases has become a crucial logistic activity. From an Operations Research perspective, managing efficiently and effectively this activity is not an easy task. Requests arrive dynamically during the day and must be served within predetermined time windows. Although previous information could be collected, it might be hard to respond timely to peaks of requests at certain hours of the day. Optimization plays thus an important role in achieving affordable logistic costs.

*The results of this chapter appears in: Côté, J. F., de Queiroz, T. A., Gallesi, F., & Iori, M. (2023). "A branch-and-regret algorithm for the same-day delivery problem". *Transportation Research Part E: Logistics and Transportation Review*, 177, 103226.

Historically, this type of problem has been addressed in the field of *stochastic dynamic vehicle routing problems* (SDVRPs), for which a broad and wide literature is known (e.g., Pillac et al. [73] and Ritzinger, Puchinger, and Hartl [78]). In recent years, the interest in such problems has risen even more, and a particular effort has been put on applications where requests have very strict time windows, of typically one hour or less. Such applications appear in dedicated online services for purchase and delivery of parcels (e.g., Amazon Prime Now) or meals (e.g., Uber Eats). Innovative optimization techniques have been consequently developed to tackle these challenging problems (e.g., Ulmer [88]).

In this context, Voccia, Campbell, and Thomas [94] introduced the *same-day delivery problem* (SDDP), a SDVRP in which requests from a limited geographical area arrive dynamically during the day and each must be served within a strict time window. All goods are based at a central depot where a fleet of identical vehicles is available to perform the deliveries. Requests that cannot be handled by the fleet are passed to a *third-party logistic operator* (3PL). The aim is to minimize the number of unserved requests, supposing that previous stochastic information on the arrival process is available and can be used to help decisions when a request is issued. Formally speaking, the SDDP corresponds to a dynamic vehicle routing problem with stochastic customers and multiple delivery routes per vehicle. Clearly, the problem models a variety of applications, including the distribution of online purchases.

The SDDP shares with other SDVRPs a number of difficult questions that a decision maker should consider when planning the distribution service, for instance: How many vehicles do we need to perform the requests efficiently? How can we efficiently route the vehicles? Should we wait for new requests, or should we start delivering as soon as possible? Can we make use of information from the past to devise better routing plans? Should we dynamically reroute the vehicles when new requests arrive? How can we effectively estimate the cost of a routing plan, given the limited information we have?

In this paper, we propose a solution approach that can help a decision-maker to obtain proper answers to the above questions in the context of the SDDP. We investigate the problem of minimizing the number of rejected requests and, as a secondary objective, the total routing cost. We do not allow transshipments among vehicles, but allow vehicles to wait at the depot to anticipate future requests. Consistently with Voccia, Campbell, and Thomas [94], we consider hard time window constraints and assume that requests impossible to deliver on time are assigned to a 3PL. The use of a 3PL is largely adopted in the related literature and is consistent with real-world applications (see, e.g., Amazon Flex, `https://flex.amazon.com/`).

More in detail, our contributions consist of:

1. An innovative *branch-and-regret* (B&R) heuristic. The B&R is equipped with four main innovation elements, namely:

   (i) New subproblem modeling. We present a new way to model the SDDP as a pickup-and-delivery problem with time windows and release dates, which brings advantages when considering preemptive vehicle returns to the depot and when planning efficient routes containing real and fictive requests from sampled scenarios;

   (ii) New scenario generation policy. We propose a new policy to generate scenarios, called *Correlated-data Sampling* (CDS), where we incorporate only fictive requests having a release time lower than the farthest end time window of any known request plus a constant value. This policy allows

the B&R to obtain improved computational results with respect to the standard policy used in the literature;

(iii) New consensus functions. We develop two new consensus functions, *Assignment Similarity* (AS) and *Edit Distance* (ED), the first of which produces better results than the consensus function *Route Similarity* (RS) used by Voccia, Campbell, and Thomas [94] and commonly adopted in the literature;

(iv) New branching scheme. We present a new branching scheme, called *Go-Now-Wait*, where we consider as branching alternatives a vehicle departing now, a vehicle departing in the future, and the assignment of a request to the 3PL. This scheme produces better results in our experiments than an adaptation to the SDDP of the branching scheme by Tirado et al. [87];

2. Extensive computational tests on benchmark instances. We conduct a comprehensive and extensive set of computational experiments that prove the outstanding performance of the B&R on benchmark SDDP instances over other classical algorithms from the literature such as the reoptimization heuristic and the *scenario-based planning approach* (SBPA). Indeed, the best algorithm by Voccia, Campbell, and Thomas [94] requires on average 96 seconds of execution time per event, while the B&R requires on average just 1 second on a similar computer, and at the same time improves by almost 20% the number of served requests;

3. Study of the preemptive depot return policy. We computationally evaluate the important problem variant in which the *preemptive depot return* (PDR) of the vehicles to the depot is allowed. In the PDR variant, vehicles are allowed to return to the depot after serving a request even if they still have items on board. This possibly allows collecting new requests and optimize the route the vehicle will perform after departing again from the depot. We show that this policy has strong managerial implications because it can lead to important cost savings, but only in the presence of particular problem conditions, such as randomly dispersed time window start times and limited fleet size.

The remainder of the paper is organized as follows. In Section 4.2, we discuss the literature on the SDDP and related problems, pointing out the main previous contributions and the similarities/differences from our work. Section 4.3 contains a formal description of the SDDP and the formal modeling of the SDDP as a dynamic pickup and delivery problem. Section 4.4 contains the details of our B&R algorithm. Section 4.5 is devoted to the computational study, with detailed results for diverse sets of parameter configurations and problem instances. In Section 4.6, we give some concluding remarks and point toward future, promising research directions.

4.2 Literature Review

This section presents a literature review focused on SDDPs arising in the context of e-commerce. For a more exhaustive review on SDVRPs, we refer to Pillac et al. [73] for a general overview, to Ritzinger, Puchinger, and Hartl [78] for a survey and a comparison of a broad range of methods from the literature, and to Ulmer et al. [89] for a study on the different techniques used to solve SDVRPs.

The SDDP shares characteristics with several well-known problems in the literature. Typically, vehicles perform several short trips from the depot to the customers

when capacity or time is limited. This feature appears in the *multi-trip vehicle routing problem* (MTVRP) and was introduced by Fleischmann [34], who proposes a savings algorithm to build routes and a bin packing heuristic to combine the built routes into work shifts. Several other works, including Battarra, Monaci, and Vigo [10], Azi, Gendreau, and Potvin [8] and Cattaruzza et al. [19], propose more complex heuristics. Exact algorithms based on column generation are proposed by Azi, Gendreau, and Potvin [6], Mingozzi, Roberti, and Toth [67] and Paradiso et al. [72]. A survey on MTVRPs can be found in Cattaruzza, Absi, and Feillet [18].

Another relevant SDDP characteristic is related to the *release date* of the requests. This represents the moment in which the requested merchandise becomes available for delivery. This implies that a request can only be part of a route that departs later than the request release date. Vehicles can then perform new routes by returning to the depot when new requests become available. This is particularly relevant in problems where requests are not all known at the beginning of the time horizon. This characteristic is studied by Arda et al. [3] in a production and transportation problem. In their study, stochastic information on release dates is used within several heuristics to solve a stochastic optimization model over a rolling horizon. Cattaruzza, Absi, and Feillet [17] consider the case where a warehouse is receiving loads of merchandise by trucks all day long. Once a load has been prepared for shipping, a route can be planned to perform the deliveries. The authors propose a genetic algorithm and solve instances with up to 100 customers.

The earliest studied applications of SDDPs can be found in the domain of e-groceries. Lin and Mahmassani [59] provide answers to several matters that are still studied today: the impact of different time window sizes, the effect of city configuration, and the delivery from either a centralized warehouse or from several grocery stores. They find out that narrow time windows are economically viable when the demand is high as this minimizes the idle time of the vehicles, whereas a low demand should be coupled with larger time windows. Their study diverges from our application as customers must order before a cut-off time. Once the cut-off time has passed, vehicles leave the depot to perform their routes, one per vehicle. Liu et al. [61] study again the SDDP for grocery delivery in the context of driverless delivery robots. They present a hybrid artificial immune algorithm and test it on different data sets, including a real one, obtaining computational good results.

For the delivery of short life span products, Azi, Gendreau, and Potvin [7] consider the case where customers are not known at the beginning of the day but are gradually revealed as time goes on. These products cannot stay on board the vehicles for longer than a specific time. This imposes the vehicles to perform several short routes, instead of a single long one. When a new event is triggered, the current known information is used to plan new routes that serve the highest revenue requests.

Voccia, Campbell, and Thomas [94] study the SDDP in the context of online purchases where all requests arrive dynamically over the course of the day. A vehicle fleet is available to pick up the goods from the depot and deliver them to the customers. Requests are typically associated with short time windows, and the objective is to maximize the number of served requests regardless of the distance traveled by the fleet. They model the problem as a Markov decision process and propose an SBPA to obtain a route plan at each decision epoch. The approach uses sampled scenarios of future requests to build a set of route plans. A *consensus function* is used to select the route plan that shares the most characteristics with the others. Such an approach was originally proposed by Bent and Van Hentenryck [11] for the dynamic

vehicle routing problem with time windows. Results indicate that more customer requests can be delivered using sampled scenarios than using a simple reoptimization heuristic.

The *dynamic dispatch waves problem* by Klapp, Erera, and Toriello [53] shares several characteristics with the SDDP. In this problem, requests do not have time windows, and a single vehicle is available to perform the deliveries. The vehicle can depart from the depot only at specific times, called *waves*, and must be back at the depot before a specific deadline. A first variant of the problem was analyzed in Klapp, Erera, and Toriello [54], who limited their study to the case in which the customers are located on a line and the vehicle operating times and costs depend only on the distance between points. Then, Klapp, Erera, and Toriello [53] addressed the problem on general network topologies, by proposing two different approaches to obtain a priori solutions. In an a priori solution, routes are built in a first stage without knowing all the information. Then, in a second stage, they are updated whenever new information is revealed by means of a recourse policy.

The *delivery dispatch problem* is closely related to the dynamic dispatch waves problem and to the SDDP. It was proposed by Minkoff [68] in the context of replenishing the inventory of a set of customers with stochastic demands. The inventory level is revealed once a vehicle becomes available to be dispatched. Routes are then planned to replenish the inventories in such a way that transportation, inventory, and out-of-stock costs are minimized. Several simplifications are made in order to model the problem as an Markov Decision Process (MDP), and a heuristic is then developed to solve it. Heeswijk, Mes, and Schutten [43] considers a similar problem faced by an urban consolidation center that dynamically receives orders from a set of customers. This work goes many steps further from Minkoff [68], by considering time window constraints and removing many simplifications. An approximate dynamic programming (ADP) algorithm is proposed and used to solve instances with up to 25 customers and a time horizon of 10 periods.

Archetti et al. [2] study the dynamic traveling salesman problem with stochastic release dates, which is similar to the dispatch wave problem. It differs from it because the time horizon is not bounded and the objective is to minimize the total travel time plus the waiting time at the depot. As noted by the authors, this type of problem might be encountered in situations where goods need to be shipped to the depot and the transportation time might be affected by traffic or other unforeseen events. The results they obtained indicate that reoptimizing at short intervals while the vehicle is waiting is beneficial for reducing the objective function.

An important issue in the SDDP is to decide whether a vehicle should be waiting at the depot for new incoming requests or should depart as soon as possible to perform deliveries and return to the depot at an earlier time to accommodate more requests later on. Waiting strategies play a crucial role in SDVRPs as they can help serve more customers and reduce the traveled distances. Some strategies only use the information known at the time the decision is taken, whereas others use stochastic information to try to predict new incoming requests. Mitrović-Minić and Laporte [69] provide four waiting strategies that do not use any knowledge about future events. The first two are opposite strategies: the *Drive-First* is a no-wait strategy where the vehicle departs as soon as possible, whereas the *Wait-First* imposes to wait whenever possible. Routes obtained from a deterministic approach that does not make use of a waiting strategy are typically those obtained by applying the Drive-First strategy, like those in Azi, Gendreau, and Potvin [7]. The other two strategies from Mitrović-Minić and Laporte [69] are in-between the previous two, and attempt to assign an amount of waiting time to some key moments of the routes. Results

indicate that waiting typically produces shorter routes but increases the number of used vehicles, whereas the no-wait strategy results in the use of fewer vehicles but at an increase in traveled distances. A different waiting strategy is proposed by Ghiani et al. [39]. Given a route, they attempt to improve it by inserting different waiting times at the nodes, but just focusing on times that are multiple of a given quantity (e.g., 5 or 10 min).

Waiting strategies are also found in Voccia, Campbell, and Thomas [94]. In their Wait-First strategy, the amount of waiting time is calculated as the maximum delay that can be added to the routes that are about to start while respecting the time window constraints. A second strategy is somehow hidden in the details of the SBPA. Indeed, as noted by Bent and Van Hentenryck [12], the SBPA has an implicit waiting strategy when real requests are assigned to a route that also contains fictive requests. In the context of the SDDP, the fictive requests indicate that new requests might arrive in the near future and the vehicle should be waiting for some time before delivering the real ones. This means that some vehicles might be waiting at the depot for new requests, while others apply a Drive-First strategy and depart as soon as possible. Results in Voccia, Campbell, and Thomas [94] indicate that the SBPA with the Wait-First strategy requires 2.8 times more computation time than the SBPA without it and cannot serve more requests.

In Bent and Van Hentenryck [12], an SBPA is used to analyze the solutions of the sampled scenarios and decide whether or not a vehicle should wait at its current location for possible new requests. This is achieved by counting the number of times a vehicle has as next visit either a real or a fictive request in the solutions of the sampled scenarios. The vehicle waits if the resulting number is higher for the fictive requests. Results show that such a strategy was helpful at maximizing the number of served customers. In our approach, at a certain epoch, we firstly optimize scenarios using a heuristic, and secondly optimize again with the same heuristic but forcing a waiting time for all the vehicles that are at the depot. If the cost of the second attempted option is smaller or equivalent to that of the first option, then we simply wait for the next event. This has the advantage of producing a much quicker computation. Otherwise, we proceed in a B&R fashion: we consider each request in the pool, evaluate if it has to be served now, served later, or rejected. For each alternative, the evaluation is obtained by building a solution with a heuristic. This is performed for all scenarios and all alternatives, and then the alternative giving the lowest average cost is chosen and implemented. This scheme has the advantage of being adaptable to different problems.

Another feature of SDDPs concerns allowing or not vehicles to perform PDRs. A PDR implies that a vehicle interrupts its current route and returns to the depot to pick up new requests. This might allow serving more requests and reducing traveled distances. The PDRs are not allowed in Azi, Gendreau, and Potvin [7], nor in Voccia, Campbell, and Thomas [94]. The idea was introduced by Ulmer, Thomas, and Mattfeld [92], who exploit it inside an algorithm based on approximate dynamic programming for the solution of single-vehicle instances. Their results show that PDRs can effectively allow more requests to be served.

In some applications, it might be required to decide immediately if a new request is accepted and delivered or if it is rejected. This requirement can be useful when some work has to be performed to make the request ready for shipment, and hence rejecting it at a later stage might cause unnecessary costs. This is defined *request acceptance policy* in Klapp, Erera, and Toriello [55] or *customer acceptance problem* in Ulmer and Thomas [90]. To our knowledge, this policy was first considered in Azi, Gendreau, and Potvin [7]. In this work, when facing a new request, the authors

first check if the request can be feasibly inserted in the current routes. If no feasible insertion position is found, then the request is rejected, otherwise, a lookahead algorithm, similar to an SBPA, is executed. This request acceptance policy is also studied in detail in Klapp, Erera, and Toriello [55] for the dispatch wave problem. They use approaches similar to those of Klapp, Erera, and Toriello [53] and calculate that an immediate request acceptance policy increases costs by an average of 4.5%.

Our approaches for solving the SDDP rely on sampling scenarios that incorporate stochastic knowledge of future events. Sampled scenarios are generated each time a new event occurs and an optimization phase is executed next to obtain a routing plan. This consists of finding a plan that can be implemented in all scenarios, leading to low-cost solutions. Typically, each scenario is solved separately to alleviate the computational complexity. Unfortunately, each routing plan is specifically tailored for its scenario and implementing one of them does not ensure achieving a low-cost solution in the other scenarios. The literature has come up with different approaches for addressing this problem, which can be seen as partial explorations of a branch-and-bound tree for a stochastic integer program (see Haneveld and Vlerk [41]). Scenarios are solved at each node of the tree to fix some first-stage decisions. For example, the SBPA proposed by Bent and Van Hentenryck [11] solves each scenario and selects the plan having the most parts in common with other plans by using a consensus function. This can be interpreted as solving only the root node of the tree.

Another strategy brought up by Løkketangen and Woodruff [62], called *progressive hedging heuristic* (PHH), and later used by Hvattum, Løkketangen, and Laporte [46] among others, is to solve all scenarios and then find the alternative a that is the most common alternative among the alternatives that are not performed in all the resulting solutions. This alternative is then fixed and plans, not having it, are optimized. The process iterates until all plans implement the same set of alternatives. Again, this can be seen as solving a node in a branch-and-bound tree, selecting a variable, and creating a single child node.

Hvattum, Løkketangen, and Laporte [47] propose the B&R heuristic as an improvement of the PHH that consists of evaluating the cost of performing or not an alternative a. The least-cost alternative, a or not a, is then fixed, and the algorithm iterates until all plans implement the same alternatives. Their results indicate that the B&R heuristic is superior in terms of solution quality to the PHH. The branch-and-bound tree, in this case, is partially explored: a node is solved, a variable is chosen for branching, and child nodes are generated and solved. The exploration continues by adopting the least-cost child node until a solution is reached. In Section 4.4 below, we detail several innovative contributions to the B&R literature.

Table 4.1 gives a brief summary aimed at contrasting our work with the main ones in the SDDP literature, both in terms of solution method adopted and characteristics of the problem solved. The most interesting contrast is with the work of Voccia, Campbell, and Thomas [94]. Both our work and theirs use stochastic information to solve the SDDP with multiple vehicles and 3PL, and, in addition, perform tests on the same instances. However, while Voccia, Campbell, and Thomas [94] model the subproblem as a *multi-trip team orienteering problem with time windows* (MTTOPTW), we model it as a *pickup and delivery problem with time windows and release dates* (DPDP). Furthermore, we accept PDRs, thus searching in a larger solution space. In addition, they use only a consensus function and a way to generate scenarios taken from the literature (described below in Section 4.3), whereas we use a new scenario generation policy and a new consensus functions. The most notable difference is that they

Table 4.1: Comparison with SDDP literature

	Multiple vehicles	PDR	Solution method	Consensus function	Branching	3PL	Stochastic information	Instance size
Archetti et al. [2]			Reoptimization				✓	50
Azi, Gendreau, and Potvin [7]	✓		Reoptimization, PHH			✓	✓	72-144
Dayarian, Savelsbergh, and Clarke [27]			Reoptimization			✓		up to 425
Dayarian and Savelsbergh [26]	✓		Reoptimization, PHH				✓	N/A
Klapp, Erera, and Toriello [53]			A priori policy			✓	✓	50
Klapp, Erera, and Toriello [54]			A priori policy			✓	✓	5-100
Klapp, Erera, and Toriello [55]			A priori policy			✓	✓	50
Minkoff [68]	✓		ADP				✓	10
Ulmer [88]	✓		ADP			✓	✓	60-180
Ulmer, Thomas, and Mattfeld [92]		✓	ADP			✓	✓	30-100
Heeswijk, Mes, and Schutten [43]	✓		ADP				✓	3-50
Voccia, Campbell, and Thomas [94]	✓		SBPA	RS		✓	✓	48-192
This work	✓	✓	B&R	AS	✓	✓	✓	96-192

use an SBPA, so they do not perform any branching scheme, which is, instead, the key ingredient of our B&R algorithm.

4.3 Problem Definition and Modeling

The SDDP considers a complete directed graph $G = (L_0, A)$, where L_0 comprises a depot, vertex 0, and a set L of customer locations distributed over a geographical area. The depot is equipped with a fleet of M identical vehicles and is associated with start ($t = 0$) and end ($t = T$) times between which vehicles can depart and arrive. The time interval $[0, T]$ corresponds to the working hours of the depot. With each arc $(i, j) \in A$ are associated a deterministic travel time t_{ij} and a traveling distance c_{ij}, which are known in advance. During the time horizon, requests arrive at a rate $\lambda_i \geq 0$ from each location $i \in L$.

Let R be the set of requests that occur during the daily time horizon. Set R is composed of requests that are known in advance (from before the starting time of the operations) and others that will be revealed as time unfolds. Each request $k \in R$ is revealed at a release time r_k and a delivery time window $[e_k, l_k]$. Each request should be picked at the central depot and delivered by a vehicle. In case a vehicle arrives at a customer location for delivering request k before the start time e_k, it waits until e_k to start the service. The service must begin before l_k, so we consider *hard time window* constraints. In addition, we consider the service time to be always equal to zero. Requests found impossible to deliver on time are assigned to a 3PL paying an additional cost. We assume that the delivery costs incurred by the fleet for performing a request are always lower than the cost of the 3PL operator.

Each vehicle may perform multiple routes during the time horizon, starting at any time $t \geq 0$ and finishing at any time $t \leq T$. The routes performed by the vehicle may involve the following actions:

(i) wait at the depot for new requests;

(ii) pick up at the depot one or more requests; and

(iii) deliver one or more requests to customers.

Once a vehicle departs from a location, it cannot divert its path until it has reached its next location. After a delivery, a vehicle can continue its route as planned or interrupt the route and return to the depot. This means the vehicle is not required to finish serving all its onboard requests before returning to the depot. In other words, we allow *preemptive returns* to the depot. Once a vehicle has picked up a request, it must serve it, so it cannot unload any request at the depot. In other words, we forbid *transshipments* among vehicles. When using the 3PL for delivering a request, we simply assume we pay a high cost and we do not explicitly model the 3PL routes. The decision of assigning a request to the 3PL is postponed until we detect that the fleet will not be able to deliver such a request.

The objective of the SDDP is to determine the routes performed by the vehicles during the time horizon, aiming first at maximizing the number of served requests and secondly at minimizing the total traveled distance.

We model the SDDP as a *dynamic pickup and delivery problem with time windows and release dates* (DPDP). Under this representation, each request $k \in R$ corresponds to a pair of nodes (p_k, d_k), where p_k is the pickup node and d_k the delivery node. In our case, p_k is always coincident with the depot 0, whereas d_k is associated with a customer location in L. Let N be the set of all the pickup and the delivery nodes, and $V = \{0\} \cup N$. Let $k(i)$ be the request associated with node $i \in N$. The time window $[e_p, l_p]$ of the pickup node p_k is set to $[r_k, l_k - t_{0k}]$, whereas that of the delivery node d_k is set to $[e_d, l_d] = [e_k, l_k]$, for each $k \in R$. In addition, let $A^+(i)$ and $A^-(i)$ be the sets of incoming and outgoing arcs from node $i \in V$.

The use of a DPDP representation of the problem gives some advantages. First, it is easy to model a preemptive return to the depot, as one simply needs to insert a pickup in the middle of two deliveries. Second, the constraint forbidding the unloading of already picked up requests at the depot is naturally taken into account by the classical DPDP constraint that imposes a delivery node to be visited by the same vehicle that visited the corresponding pickup node. Other papers have made similar modeling approaches for on-demand same-day delivery and dial-a-ride settings, see, e.g., Arslan et al. [4] and Ulmer et al. [89]. Other authors, as Voccia, Campbell, and Thomas [94] and Archetti et al. [2], modeled each request of the SDDP as a single delivery node located at a customer location. This has the advantage of being faster to optimize, but it is less flexible and might remove some sequencing possibilities. For example, in Voccia, Campbell, and Thomas [94] if a fictive delivery (originated when modeling the stochastic component of the problem) is visited by a vehicle right after a real delivery, then the vehicle is sent back to the depot. This makes routes containing real and fictive customers difficult to plan efficiently.

We tackle the dynamic aspect of the SDDP in the classical SDVRP approach (see, e.g., Gendreau et al. [38]): each time a new event occurs, all the known information is gathered together, and an optimization step is executed. This step requires making several decisions (e.g., if a vehicle departs from the depot or another performs a preemptive return). This classical approach may change consistently according to the way events and optimization are taken into account. During the working period, an *event* occurs each time new information becomes known, or new decision has to be taken. In our work, we consider three types of events:

- arrival of a new request when there is at least one vehicle available at the depot;
- arrival of a vehicle at the depot or completion of the waiting period of a vehicle (still at the depot);

- completion of a delivery by a vehicle that should then visit another customer (and not return directly to the depot).

The first two types of events are the only ones considered in Voccia, Campbell, and Thomas [94]. The third type is used only when allowing preemptive returns to the depot.

When an event occurs, all known information is collected (known requests, the position of the vehicles, goods that are on board the vehicles, etc.), and an optimization algorithm is invoked to take the next routing decisions. The optimization algorithm returns a routing plan each time it is invoked. A *routing plan*, or just *plan* for short in the following, is a partial solution to the dynamic problem, which might be later modified/integrated according to new information revealed. Note that some papers do not use the term solution when referring to a routing plan, but *policy*, so as to give more emphasis to the dynamic aspect of the problem. In this paper, we adopt the term solution.

Being dynamic problems, SDDPs are mathematically modeled as Markov Decision Problems (MDP). A formal MDP has been presented by Voccia, Campbell, and Thomas [94]. In the following, we present, instead, a mathematical model of the static deterministic case, when everything is precisely known in advance, based on integer linear programming. The decision variables are the following ones:

- x_{ij}^m is equal to 1 if arc (i,j) is traversed by vehicle m, 0 otherwise;
- y_{km} is equal to 1 if request k is served by vehicle m, 0 otherwise;
- S_i indicates the starting time of service at node $i \in V$.

The model can then be formulated as follows:

$$\min \ (|R| - \sum_{m\in M}\sum_{k\in R} y_{km})U + \sum_{m\in M}\sum_{(i,j)\in A} c_{ij}x_{ij}^m \tag{4.1}$$

$$\text{s.t.} \sum_{j\in A^+(i)} x_{ij}^m = y_{k(i)m} \qquad m \in M, i \in N \tag{4.2}$$

$$\sum_{j\in A^-(i)} x_{ji}^m = y_{k(i)m} \qquad m \in M, i \in N \tag{4.3}$$

$$\sum_{j\in A^+(0)} x_{0j}^m \leq 1 \qquad m \in M \tag{4.4}$$

$$\sum_{m\in M} y_{km} \leq 1 \qquad k \in R \tag{4.5}$$

$$S_j \geq S_i + (t_{ij} + U)\sum_{m\in M} x_{ij}^m - U \qquad (i,j) \in A \tag{4.6}$$

$$e_i \leq S_i \leq l_i \qquad i \in V \tag{4.7}$$

$$S_{d_k} \geq S_{p_k} + t_{p_k d_k} \qquad k \in R \tag{4.8}$$

$$x_{ij}^m \in \{0,1\} \qquad (i,j) \in A, m \in M \tag{4.9}$$

$$y_{km} \in \{0,1\} \qquad k \in R, m \in M \tag{4.10}$$

$$S_i \geq 0 \qquad i \in V. \tag{4.11}$$

The objective function (4.1) minimizes the unserved requests and the overall transportation cost, with U being a large number. Constraints (4.2) and (4.3) ensure the two nodes of the request are visited by the same vehicle and also impose degree constraints on these nodes. Constraints (4.4) ensure that at most M vehicles are used. Constraints (4.5) force a request to be served by at most one vehicle. Constraints

(4.6) and (4.7) guarantee feasibility with respect to time windows. In addition, constraints (4.6) also eliminate subtours. The precedence order is preserved by means of constraints (4.8). Constraints (4.9), (4.10) and (4.11) impose the domain of the variables.

4.4 Branch-and-regret Algorithm

In this section, we present the detail of the B&R heuristic that we implemented for the SDDP. The B&R builds upon the classical reoptimization heuristic and SBPA, which we briefly revise here.

The *reoptimization heuristic* is a dynamic algorithm that ignores the stochastic aspects of the problem and works as follows: first, an empty *routing plan* s is created and all known requests at the beginning of the time horizon are added to s; second, s is optimized by a heuristic; then, each time there is a new event, all actions that were performed prior to the current time and all those being currently performed are locked in plan s, while the possible newly revealed requests are added to s and another call to the optimization algorithm is performed to adjust s. Once all events have been considered, the algorithm terminates with a final plan containing all routing actions.

The SBPA (Bent and Van Hentenryck [11]) operates as the reoptimization heuristic, but at each event, instead of directly computing a routing plan s with the known requests, creates a set of scenarios Ω containing fictive requests. Formally, a *scenario* is a set of customer requests including both known requests, which have been already issued by the customers, and *future requests* (also called *fictive requests*), which might appear later on and are generated by sampling the probability distribution of their appearance. In detail, at each event the SBPA creates a plan s_ω for each scenario $\omega \in \Omega$, including all known requests and locked components of s, plus the set of fictive requests from ω. Each plan s_ω is optimized by invoking a heuristic, and the routes having at least one fictive request are removed. This removal can be seen as a way to delay the departure of the corresponding vehicles to possibly accommodate new requests that might arrive in the near future. At last, all plans are evaluated by means of a consensus functions (see Section 4.4.2 below) and the plan with the highest consensus function score is selected.

4.4.1 Branch-and-Regret Heuristic

The B&R heuristic was proposed by Hvattum, Løkketangen, and Laporte [47] for solving SDVRPs. Its main components are similar to those adopted in the SBPA, but the method goes a step forward to find solutions that should be better on average. In the B&R, at each new event, scenarios are generated and plans are obtained by optimizing the scenarios. Next, the costs of implementing different alternatives are evaluated with the aim of finding a plan that can be implemented in all scenarios. In Hvattum, Løkketangen, and Laporte [47], this is achieved in two steps: first, assign a narrower time window to each known request, and second, select which requests are served next by the available vehicles. The average cost of a narrower time window for a specific request is obtained by fixing the time window to be in the next time interval in each scenario. The average cost of *regretting* this alternative is also calculated by changing the time window to a farther time in the future. The *branching* consists in imposing the time window leading to the least average cost. The second step, aimed at selecting which requests are served next, is performed once

the time windows of all known requests have been fixed. Once this two-step decision phase is concluded, the actions to be performed immediately are implemented, whereas the other actions, like the time window changes, are canceled (as they can be reevaluated at the next event).

This branching scheme is better suited for problems involving only pickup operations. The routes in these problems are not decided in advance, they are built gradually as customer requests arrive throughout the day. For the SDDP, all customers of a route have to be figured out before the vehicle can depart. We thus differ from Hvattum, Løkketangen, and Laporte [47] by evaluating different alternatives that are valuable in the context of the SDDP. This is done in three steps:

1. evaluating if the vehicles at the depot should wait;
2. ensuring that the same alternative is implemented in each scenario for each known request;
3. selecting a plan with a consensus function.

In the first step, we evaluate two alternatives for the whole fleet: (a) vehicles at the depot perform their routes as planned, and (b) vehicles at the depot wait for one unit of time. In practice, we first optimize all scenarios without explicitly allowing vehicles at the depot to wait, and we compute the average cost. This gives the cost of alternative (a). For alternative (b), we try to postpone for one unit of time every route that is about to leave in the solution of each scenario. If this is feasible, we compute the cost of this scenario. Otherwise, we reoptimize the solution of the scenario to force all vehicles at the depot to wait for one unit of time. The average cost is computed to get the cost of alternative (b). If the second alternative is not more expensive than the first one, then we opt to wait. To this aim, we create a new event whose delay is set to the maximum delay that can be added to the routes of the scenarios so that they remain feasible. If, instead, the second alternative is more expensive than the first one, then we proceed to the second step.

In the second step, we make sure that for all known requests, exactly one among the three following alternatives is selected in all scenarios:

- a vehicle departs now from the depot to deliver the request (*go now*);
- a vehicle departs from the depot at a future time to deliver the request (*wait*);
- the request is not delivered at all (*reject*).

If at least one request does not implement the same alternative in all scenarios, we iterate through these *unfixed* requests to ensure that all plans implement the same alternative for each request. At each iteration, we select the unfixed request having the highest count of the *go now* alternative. Next, we evaluate the average cost of performing each of the previous three alternatives. This is done by calculating the routing cost of imposing each alternative in each scenario. The alternative having the lowest average cost is chosen and implemented. This evaluation process is performed until each known request implements the same alternative in all scenarios.

Finally, the third step is to choose a plan using a consensus functions (to be defined in Section 4.4.2).

A pseudo-code of our B&R heuristic is presented in Algorithm 1. At each event, the routing plan s is locked, the new requests are added to s, and a set of scenarios is generated. Each scenario ω is optimized to obtain a plan s_ω. We evaluate the alternative of having the vehicles waiting at the depot. If this is not more expensive than

having the vehicles performing their routes as planned, we wait for the next event. Next, we calculate from the plans s_ω the set L of requests that do not implement the same alternative in all scenarios. Next, we perform the following steps until L is empty. First, for each request $r \in L$, we calculate $gonow[r]$ as the number of times r is in a route that departs now in the plans s_ω. We select the request r having the highest $gonow[r]$ value. We define Φ_r as the set of alternatives for request r and for each alternative $\phi \in \Phi_r$ we impose ϕ in each scenario ω to obtain the plan s_ω^ϕ. We calculate the average cost of each alternative, and the alternative ϕ with the lowest cost is chosen and implemented ($s_\omega = s_\omega^\phi$). Set L is then updated using the plans s_ω, and the process is iterated for the next request. Once L is empty, we implement the plan s_ω with the highest score, that is, we remove the future requests from s_ω to obtain s.

Algorithm 1 Branch-and-Regret heuristic for the SDDP

Input: Consensus function f

Create a plan s that contains the requests known at time 0
while there is an event or time = 0 **do**
  Lock all performed actions in plan s
  Add the new requests to s
  Generate scenario set Ω of fictive requests
  for scenario ω in Ω **do**
    Set $s_\omega = s \cup \omega$ and optimize plan s_ω with ALNS
  end for
  Evaluate if the vehicles at the depot should wait, if so, wait for the next event
  L: the requests that do not implement the same alternative in all scenarios
  while L is not empty **do**
    Calculate $gonow[r]$ from plans s_ω for each $r \in L$
    Select the request $r = \arg\max_{r \in R'}\{gonow[r]\}$
    for each alternative ϕ in Φ_r and scenario ω in Ω **do**
      Set $s_\omega^\phi = s_\omega$ and optimize s_ω^ϕ with ALNS imposing ϕ on r
    end for
    Implement the least-cost alternative ϕ on r
    $s_\omega = s_\omega^\phi$
    Update L from the plans s_ω
  end while
  Select the plan s_ω having the highest score on consensus function f
  $s = s_\omega \setminus \{\omega\}$
end while

4.4.2 Consensus functions

The B&R, as well as the simpler SBPA, selects the routing plan to be adopted by using a *consensus function*. A consensus function is a function that receives in input the set of routing plans generated for all scenarios and then returns a score for each of them. The plan with the highest score is the one adopted for implementation. Voccia, Campbell, and Thomas [94] use a function, which we name *Route Similarity* (RS), that consists of counting the number of times the routes of a plan appear in other plans. To better describe RS, and other new consensus functions that we propose here, we make use of the following example.

Example 1. We are given three routing plans (i.e., solutions), each having three routes that can possibly depart immediately. By defining a plan in square brackets, and a route in round brackets, the example is: [(1-3-5-7), (4), (6-2)], [(1-3), (4-7), (6)] and [(wait), (4), (6)].

In Example 1, the routes of plans #1 and #2 appear just once, whereas those of plan #3 appear twice each, and hence plan #3 would be the one chosen and implemented by the RS function. The drawback of a plan built with this strategy is that it might require a larger number of routes, thus increasing the distance, without performing the most common actions found among scenarios. For Example 1, in 2 out of 3 plans, the most common actions are to depart now and deliver requests 1, 3, 4, 6, and 7.

In the following, we propose two new consensus functions that look at different actions in the plans. The first, called *Assignment Similarity* (AS), counts the number of times the pairs (request, route number) of a plan appear in other plans. Ties are broken by plan number. In Example 1, plan #1 has an AS score of 6, plan #2 of 6, and plan #3 of 4. The first plan would thus be the one selected by function AS.

The second function, called *Edit Distance* (ED), sums the number of changes required in a plan to obtain each of the other plans. The plan having the smallest number of required changes is selected. Function ED builds upon the Levenshtein distance (Levenshtein [58]), which is used to count the minimum number of changes required to change one word into another word. For Example 1, to obtain plan #2 from plan #1, we would need to remove requests 5 and 7 from route 1, add request 7 to route 2, and remove request 2 (resulting in 4 changes). To obtain plan #3 from plan #1, we would need instead to remove requests, 1, 2, 3, 5, and 7 (for a total of 5 changes). The ED score of plan #1 is thus 9. Similarly, the ED score of plan #2 is 7, and that of plan #3 is 8. The second plan would thus be selected by function ED.

As the vehicle fleet is homogeneous, an additional step is added in ED to avoid counting the number of changes between two solutions having the same routes but performed by different vehicles. For instance, let us add to Example 1 a fourth plan that is simply obtained by a rearrangement of the routes of the first plan, namely [(6-2), (1-3-5-7), (4)], then the number of changes according to the ED function would be high between plans 1 and 4 (i.e., 14 changes would be needed). To avoid this inaccurate evaluation, when comparing two plans, we create an instance of the classical *Assignment Problem* (AP), where the nodes on the left side of the bipartite graph are the M routes of the first plan, and the nodes on the right side are the M routes of the second plan. The cost of each arc is set to the number of changes required to transform a route into the other. An AP is solved for each pair of plans, through the Hungarian algorithm, to obtain the minimal number of changes regardless of the route assignments.

A function that is similar to ED was proposed by Song et al. [82] for a multi-period team-orienteering problem. They compute the similarity between two plans by using a Hamming Distance. Then, they attempt reducing symmetries by solving the AP above, but just in a heuristic way. A detailed computational assessment of the RS, AS, and ED consensus functions is provided below in Section 4.5.

4.4.3 Scenario Generation

The B&R use sampled scenarios to guide the decision process. The number of scenarios has a major impact on solution quality and computation time. Typically, computation time increases linearly in the number of scenarios. Different conclusions are

taken in the literature on the number of scenarios that produce the best trade-off between solution quality and computing effort. Hvattum, Løkketangen, and Laporte [46] tested several sizes ranging from 30 to 600 scenarios and concluded that the best option is to have as many scenarios as possible. In a different setting, Hvattum, Løkketangen, and Laporte [47] obtained a different conclusion as, after testing sizes from 1 to 60 scenarios, they obtained their best solutions with 30 scenarios. In Voccia, Campbell, and Thomas [94], the authors made tests using 10, 25, and 50 scenarios, and noted that the best performance was obtained using 10 scenarios. Clearly, all these conclusions are problem-dependent and based on the particular instances addressed in those papers. In Section 4.5.1 below, we obtain some more insights on this aspect by means of extensive tests performed by varying the number of scenarios.

The size of the sampling horizon is also a relevant aspect to solve the SDDP. Voccia, Campbell, and Thomas [94] opted to maintain in the scenarios all fictive requests that appear in the successive ρ instants of time. This reduces the subproblem size, lowers computation times, and emphasizes the decisions that have to be taken in the immediate future. The rationale is that fictive requests in a faraway future might simply act as noise in the decision-making process. This strategy is also tested in Section 4.5.1. In addition, we propose a new alternative method. As it can be noted, using a fixed sampling horizon can perform well when the length of the horizon is correlated with the data of the instance. However, lower quality results might be expected when ρ is too small and the time windows are large, as this might lead to too early departures. The alternative method that we developed, called *correlated-data sampling* (CDS), incorporates only the fictive requests having a release time lower than the farthest end time window of any known request plus a constant value $\bar{\rho}$. The idea, computationally tested below, is to consider only fictive requests that can impact the decision process.

4.4.4 Optimizing Subproblems

The B&R is based on the iterated solution of DPDP subproblems that appear during the search. This is made by an *optimize* function, invoked at steps 7 and 15 of Algorithm 1. The function receives as input a plan that might contain empty routes and not-yet assigned requests, routes with completed requests, or routes with a mix of completed and non completed requests. Its aim is to obtain a plan that satisfies all constraints of the DPDP indicated in Section 4.3 and has the minimum total cost. In addition, the function returns a high cost for any plan not respecting the decisions taken at the previous branches. The function that we implemented is based on the execution of four steps.

First, the unassigned requests are sequentially inserted inside the plan using the *Regret-k* heuristic of Potvin and Rousseau [74]. The algorithm works as follows: for each request, it calculates the minimal insertion cost of the request inside each route. Then, at each iteration, it selects the request having the largest sum of differences between the lowest insertion cost and the insertion cost into the other k best routes, in absolute value. The selected request is inserted in the route with the lowest insertion cost. The minimal insertion cost of the remaining requests inside this route is updated. If we cannot feasibly insert a request, then this request is inserted into a bank and resumed later in the local search phase.

Second, the local search operators *Relocate* and *Exchange* are executed. The two algorithms operate similarly: Relocate removes a request from its current position in a route or from the request bank, and attempts to reinsert it in another position in the same route or another one, whereas Exchange takes two requests from different

routes and tries to insert them in each other route. Both methods look for insertion positions that minimize the cost of the resulting plan. If improving positions are found, they are implemented; otherwise, the requests are reinserted in their original positions. The two operators are executed, one after the other, until no further improvement can be found.

Third, the classical ALNS of Ropke and Pisinger [79] is called. We adopt an ALNS approach because of the good results it achieved on a number of related applications (see, e.g., Sun et al. [84]). At each iteration, a removal and an insertion heuristic are selected from a pool of heuristics. A random number of requests are then removed from the plan and inserted in the request bank using the removal heuristics. Then, the insertion heuristic tries to insert them back in the plan with the hope of finding an improved solution. In addition, the removal and insertion heuristics make sure that completed or fixed requests remain in their position. Our implementation is the same as the one described in Ropke and Pisinger [79], with the following exceptions: 1) we only use the Random and Shaw removal heuristics; 2) the cooling rate is set to 0.8; and 3) the number of iterations is decided according to the tests of Section 4.5.

Fourth, the two local searches invoked at step two are invoked once more in a last attempt to improve the plan.

4.4.5 Dealing with Preemptive Depot Returns

The option of allowing or not PDRs was formally proposed by Ulmer, Thomas, and Mattfeld [92]. For many works in the literature (namely, Azi, Gendreau, and Potvin [7], Voccia, Campbell, and Thomas [94], Klapp, Erera, and Toriello [53] and Archetti et al. [2]), once a vehicle departs to perform deliveries, it has to complete its entire route before returning to the depot. PDRs allow vehicles to return to the depot before the routes are completed. Enabling this might help at reducing distances or at delivering more requests. The conclusion on whether this policy is advantageous or not are mixed. Klapp, Erera, and Toriello [53] claim that the benefits are marginal. On the other hand, Ulmer, Thomas, and Mattfeld [92] state that the policy leads to relevant savings.

Our framework can easily deal with the PDR variant because of the way the SDDP is modeled. As mentioned, we represent the SDDP as a DPDP, and having pickup and delivery nodes enable subproblems to decide easily if a pickup is to be inserted between two deliveries belonging to a newly departed route. The pickup would represent a return to the depot. Each vehicle maintains a *current node* that represents the first node after which a request can be inserted. Route modifications can only occur after that node. Note that if we want to forbid PDRs, then we simply set the current node to the last delivery node of the route.

4.5 Experimental Results

In this section, we present the results of extensive computational tests performed to evaluate the B&R. Each instance was solved 10 times, each time with a different seed. In detail, we determine the best B&R configuration (Section 4.5.1), we evaluate the impact of PDRs (Section 4.5.2), we compare our B&R against the literature (Section 4.5.3), we test it on large-size instances (Section 4.5.4), and finally study in detail the solution structures (Section 4.5.5). Our algorithm has been coded in C++ and our tests have been executed by using a single core of an Intel 2.667 GHz Westmer EP

X5650 processor. For comparison purposes, we have made the instances that we used publicly available at https://sites.google.com/view/jfcote/.

4.5.1 Parameter Setting and Sensitivity Analysis

This section analyzes the results obtained by the B&R algorithm under different parameter configurations on a set that contains 125 instances selected from the benchmark set by Voccia, Campbell, and Thomas [94] plus 25 additional instances that we created. The instances are divided into three types according to the way customer locations have been generated: clustered; randomly dispersed; and both randomly dispersed and clustered. In addition, the instances are characterized by five types of time windows. The first four types, namely TW.d1, TW.f, TW.h, and TW.r, have all a one-hour deadline but differ on the start time of the time windows, which is equal to the release date for TW.d1, a fixed time in the future for TW.f, the remaining hours of the day for TW.h, and randomly dispersed times for TW.r. The fifth type, called TW.d2, is equivalent to TW.d1 but has a two-hour deadline. An additional set of 25 instances were also created from the 25 TW.d1 instances by simply removing the time windows. These 25 instances are called *No TW* in the following. The total number of instances is thus 150. The instances are grouped according to 15 different location distributions. The arrival rate is equal to 0.002 per minute for each customer location, and there are 100 customer locations. Requests arrive during a time horizon of 480 minutes, thus producing an average of 96 requests (in our subset, the number of requests varies indeed from 71 to 110). All vehicles should be back to the depot before minute 540.

In all the tests of this section, we considered a fleet of 10 vehicles, 60 minutes of time horizon, and 30 scenarios. Moreover, waiting at the depot is allowed, but PDRs are not. In the tables below, we evaluate each algorithm in terms of:

- %filled = percentage of requests served;
- dist. = total distance traveled by the vehicles;
- time = computing time in seconds.

In Table 4.2, we evaluate the B&R by attempting different numbers of ALNS iterations (namely, 50, 100, 250, 500, and 1000) and the three consensus functions (namely, RS, AS and ED). The values shown in the table are the average of the values obtained on the 150 instances. A final line showing average values for the entire column is also reported to gain some insight into the impact of the attempted parameter on the algorithm. The rightmost column presents, instead, the average results over all tests performed with the given algorithm. The results of Table 4.2 show that with a small number of ALNS iterations the AS consensus function obtains the best %filled. When the number of iterations increases the RS obtains slightly better results. In general, RS and AS perform well, while ED has a slightly weaker performance. The use of 1000 ALNS iterations allows almost all functions to produce their best results, but this is obtained at the expense of high computing times. We note that a good trade-off between quality and time is obtained by using 250 ALNS iterations, and we kept this value in all the successive tests. We also note that at a first glance the differences between the percentage of requests served may appear to be small, but they are very relevant from a managerial point of view, as they can bring a positive advantage over competitors and eventually lead to a profitable business even when unitary margins are not high.

Table 4.2: Attempting different ALNS iterations

	ALNS It.=50			ALNS It.=100			ALNS It.=250			ALNS It.=500			ALNS It.=1000			average
algorithm	%filled	dist.	time	%filled	dist.	time	%filled	dist.	time	%filled	dist.	time	%filled	dist.	time	%filled
B&R-RS	93.41	2140.3	98.9	93.48	2132.9	136.5	93.64	2128.3	251.2	93.75	2124.3	448.6	93.78	2119.9	841.6	93.61
B&R-AS	93.46	2142.1	99.7	93.57	2134.1	137.5	93.66	2124.2	251.4	93.65	2125.4	450.8	93.78	2120.4	844.0	93.62
B&R-ED	93.37	2142.6	98.8	93.41	2137.0	135.6	93.62	2125.3	248.8	93.70	2122.0	443.5	93.75	2117.1	831.2	93.57
average	93.41	2141.6	99.1	93.49	2134.6	136.5	93.64	2125.9	250.5	93.70	2123.9	447.6	93.77	2119.1	839.0	93.60

Next, we evaluated our algorithm by attempting different time horizon strategies. The first one, proposed by Voccia, Campbell, and Thomas [94] and discussed in Section 4.4.3, works on the size of the sampling horizon by maintaining the requests that appear in the next ρ instants of time. The newly-introduced CDS strategy maintains in the scenarios only the fictive requests that have a release time lower than the farthest end time window of any known request plus a constant time $\bar{\rho}$. For Voccia, Campbell, and Thomas [94] we attempted ρ=60, 120, and $+\infty$, while for CDS we attempted $\bar{\rho}$=0 and 15. The results that we obtained are shown in Table 4.3. The columns have the same meanings as those of Table 4.2. The results show that the CDS strategy slightly improves the Voccia, Campbell, and Thomas [94] strategy, using less computing time to obtain solutions with better %filled values. However, the CDS strategy with $\bar{\rho} = 0$ reduces distances by 1.5% and time by 22.8% compare with the other strategy with $\rho = +\infty$. The best average %filled value of 95.49% is achieved for the CDS strategy and both values of $\bar{\rho}$. We thus selected $\bar{\rho} = 0$ for all the next tests. From a managerial point of view, this implies that with CDS one can obtain the best performance by considering only fictive requests whose release time is within the known request's time windows.

Table 4.3: Attempting different time horizons

	Voccia, Campbell, and Thomas [94] strategy									correlated-data strategy					
	ρ=60			ρ=120			ρ=$+\infty$			$\bar{\rho} = 0$			$\bar{\rho} = 15$		
algorithm	%filled	dist.	time	%filled	dist.	time	%filled	dist.	time	%filled	dist.	time	%filled	dist.	time
B&R-RS	93.64	2128.3	246.1	94.75	2194.3	338.3	95.46	2299.1	533.7	95.48	2263.3	411.5	95.49	2268.1	420.5
B&R-AS	93.66	2124.3	246.3	94.78	2190.8	339.7	95.48	2300.7	536.0	95.51	2265.5	413.4	95.50	2266.2	422.4
B&R-ED	93.62	2125.3	243.8	94.79	2188.6	334.1	95.45	2297.3	529.1	95.46	2263.7	408.1	95.47	2268.5	417.3
average	93.64	2126.0	245.4	94.77	2191.2	337.3	95.46	2299.0	532.9	95.49	2264.2	411.0	95.49	2267.6	420.1

The next parameter that we evaluate is the number of sample scenarios. In Table 4.4, we consider the same approaches used in the previous tables, and test each of them with a number of sample scenarios that varies from 5 to 30. We can notice that the number of scenarios positively affects the %filled values, which increase slightly but constantly. Also, the distances increase slightly, but this is due to the higher number of requests delivered. As expected, the number of scenarios has a relevant impact on the computing times. Based on the results, we opted to use 30 scenarios and the AS consensus function in all next experiments as this is the configuration that leads to the highest average %filled value.

In Table 4.5, we compare our branching scheme, called Go-Now-Wait, with an implementation of the branching scheme by Tirado et al. [87]. This scheme is the best suited scheme from the literature for the SDDP. It looks at all pairs request-vehicle, and the pair of highest frequency is chosen for branching. Two branches are then created: in the first one the request is assigned to the vehicle, whereas in the second one it is forbbiden to be assigned to the vehicle. The best results are obtained by Go-Now-Wait, which produces limited but consistent improvements on both %filled and traveled distance. Notably, the new scheme also reduces the

Table 4.4: Attempting different numbers of sampled scenarios

	Scenarios=5			Scenarios=10			Scenarios=20			Scenarios=30		
algorithm	%filled	dist.	time	%filled	dist.	time	%filled	dist.	time	%filled	dist.	time
B&R-RS	94.99	2242.4	48.2	95.18	2237.1	111.4	95.39	2257.3	256.3	95.48	2263.3	414.1
B&R-AS	94.99	2241.0	48.2	95.26	2239.5	111.5	95.45	2258.9	256.3	95.51	2265.5	416.1
B&R-ED	95.01	2240.2	48.1	95.19	2236.7	111.1	95.41	2255.2	254.8	95.46	2263.7	410.5
average	95.00	2241.2	48.2	95.21	2237.8	111.3	95.41	2257.1	255.8	95.49	2264.2	413.6

Table 4.5: Attempting different branching schemes in B&R-AS

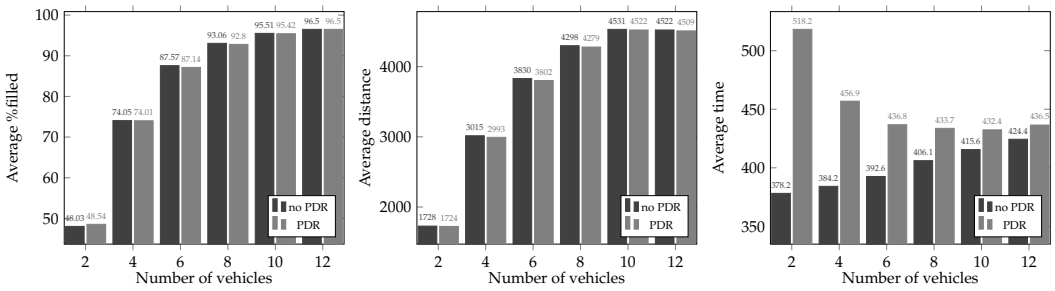
algorithm	scheme	%filled	dist.	time	routes	events	time/event	nodes
B&R-AS	Go-Now-Wait	95.51	2265.5	409.5	25.6	221.9	2.1	494.6
B&R-AS	Tirado et al. [87]	94.99	2389.4	809.9	27.4	198.1	4.4	1502.0

total computing time, by practically halving it. This can be imputed to the fact that the nodes produced by Go-Now-Wait are about one-fourth of those produced by the other scheme.

4.5.2 Evaluations of Preemptive Depot Returns

This section evaluates the gains that can be obtained by allowing or not PDR. The same 150 instances from the previous section and only the B&R-AS are used in the following experiments. The algorithm was tested by varying the number of available vehicles, attempting 2, 4, 6, 8, 10, and 12 vehicles. Figure 4.1 depicts the results when PDRs are either allowed or not. The increase obtained in %filled by allowing PDR is evident when the number of vehicles is very small. Instead, when the number of vehicles increases the PDRs have a negligible or even negative impact. In terms of traveled distance, the PDRs decrease the number of requests delivered and this automatically leads to smaller traveled distances. In terms of computing times, PDRs require a larger computing effort with respect to the case with no PDRs, but this difference decreases when the number of vehicles increases. At a first glance, it thus appears that the use of PDRs does not lead to relevant benefits. However, by deepening the analysis and differentiating it by time window types and fleet sizes, we found out that there are cases in which PDRs do help in producing improvements.

Figure 4.1: Average %filled, distance, and computing time with and without PDR



In Table 4.6, we try to obtain further insight by disaggregating the results by time window type. The table details the average %filled values obtained by B&R-AS with

and without PDRs and with different vehicle numbers. Each value gives an average over 25 instances, with the exception of the bottom line and rightmost columns that give overall average values. Best values are highlighted in bold to ease comparison. We can notice a different impact of PDRs on the time window types. For the No TW case the impact of PDRs is negative, as their use reduces the %filled values. This is due to the fact that, as there is no time constraint to satisfy, it does not pay in practice to interrupt a route and go back to the depot to collect newly released requests, but it is better to conclude the route. However, for all the other time windows the PDR is useful with a small number of vehicles. This is due to the larger need for optimization when the problem is more constrained and the vehicles are a scarce resource. When the number of vehicles increases, the PDR is no more useful. On average, for TW.d1, TW.d2, TW.f, which are all characterized by a strict time window and a somehow regular time window start time, the impact is negligible, and in a few cases, even negative. For TW.h and TW.r, which are characterized by randomly dispersed time window start times, the impact is, instead, very relevant.

Table 4.6: Impact of preemptive depot returns on %filled per time window type (2, 4, 6, 8, 10, and 12 vehicles)

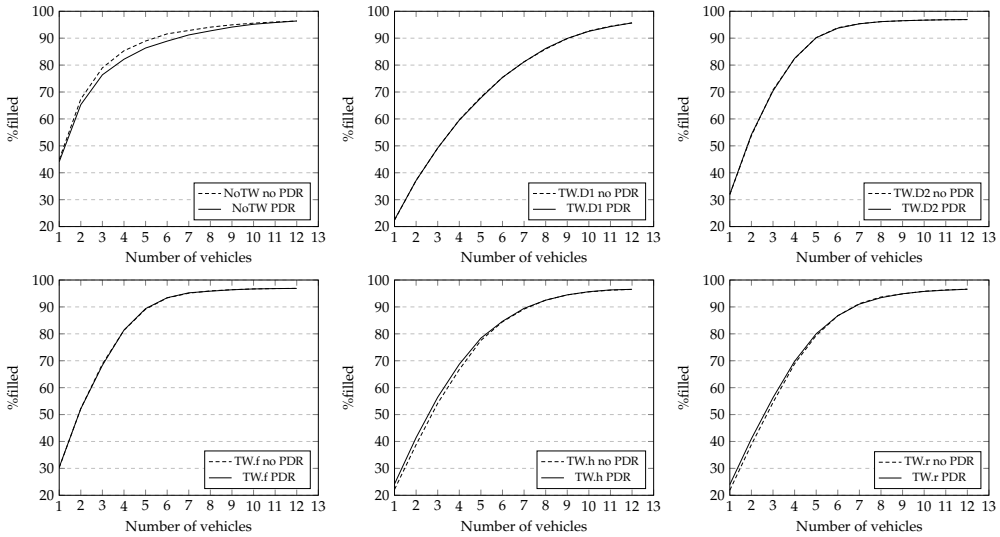
	B&R-AS													
	2		4		6		8		10		12		average	
TW type	no PDR	PDR	no PDR	PDR	no PDR	PDR	no PDR	PDR	no PDR	PDR	no PDR	PDR	no PDR	PDR
No TW	**67.37**	65.30	**85.33**	82.18	**91.61**	88.91	**94.10**	92.72	**95.58**	95.18	**96.39**	**96.39**	**88.40**	86.78
TW.d1	**37.14**	37.12	**59.65**	59.55	75.42	**75.43**	85.96	**86.18**	**92.68**	92.56	**95.68**	**95.68**	**74.42**	**74.42**
TW.d2	53.78	**54.20**	82.42	**82.48**	**93.88**	93.71	96.14	**96.15**	**96.72**	96.70	**96.92**	**96.92**	86.64	**86.69**
TW.f	**52.24**	52.17	81.33	**81.39**	93.35	**93.41**	**95.97**	95.85	**96.69**	96.68	**96.90**	96.87	**86.08**	86.06
TW.h	38.80	**41.43**	66.72	**68.70**	84.47	**84.65**	**92.53**	92.50	95.58	**95.66**	96.51	**96.54**	79.10	**79.91**
TW.r	38.84	**41.01**	68.86	**69.79**	86.71	**86.73**	**93.68**	93.43	**95.84**	95.76	**96.59**	96.57	80.09	**80.55**
average	48.03	**48.54**	**74.05**	74.01	**87.57**	87.14	**93.06**	92.80	**95.51**	95.42	**96.50**	**96.50**	**82.45**	82.40

The findings that we highlighted for 2, 4, 6, 8, 10 and 12 vehicles are confirmed also for other fleet sizes in Figure 4.2, where we report the outcome of more extensive results. Each point in the figure is obtained by solving 25 instances with a given number of vehicles (x-axis) and a given time window (subfigure), with and without PDRs. The y-axis reports the average %filled values. We attempt all numbers of vehicles from 1 to 12, so the figure summarizes the outcome of 36000 tests. We can notice that the two lines (PDR and No PDR) are almost coincident for the TW.d1, TW.d2, and TW.f cases. This confirms the fact that the impact of PDRs is negligible for instances having narrow time windows that are close to their release time. For TW.h and TW.r the use of PDRs allows B&R-AS to achieve good improvements. For the No TW case, PDR decreases the percentage of served requests, especially in the range from 2 to 9 vehicles.

A number of interesting managerial implications may be devised from the outcome of these tests. First, it is imperative for a company to push their customers in accepting wide time windows for the deliveries. Indeed, when passing from TW.d1 (one-hour time window) to TW.d2 (two-hour) we can notice a remarkable increase of about 12% in the percentage of served requests. Second, an initial effort must be spent in finding the good fleet size, as a low number (i.e., 2, 4, or 6 in our tests) may lead to many unserved requests, whereas a large number (i.e., 10 or 12 in our tests) can lead to a very limited increase in the %filled but at the expenses of a large increase in costs. Lastly, it is important to consider the structure of the time windows to decide whether PDRs are allowed or not. Indeed, in the presence of randomly dispersed time window start times and limited fleet size, it may be convenient for a

company to equip the drivers with fast communication systems so as to be able to reroute them back to the depot when needed.

Figure 4.2: %filled for the six types of time windows for the B&R-AS algorithm



4.5.3 Comparison with Voccia, Campbell, and Thomas [94]

In Table 4.7, we compare our algorithms with those of Voccia, Campbell, and Thomas [94] on their entire Hom2 set, which contains 4050 instances each having an expected number of requests equal to 96 and just three vehicles. Algorithms are sorted by increasing %filled value. We compare the results obtained by Voccia, Campbell, and Thomas [94] with their Reoptimization and SBPA-RS algorithms, with those obtained by our implementations of Reoptimization, SBPA-RS, SBPA-AS, and B&R-AS with and without PDRs. Our implementations of Reoptimization and SBPA rely on the routing optimize function of Section 4.4.4 to solve the subproblems. We also show, in "Offline", the results obtained on the static variant of the problem in which all information is assumed to be known in advance. This variant is solved with a unique call to the routing optimize function. For each algorithm, we present the values already discussed in the previous tests, in addition to the number of events (i.e., the number of times the algorithm is invoked) and the time per event (computed as time/events). For Voccia, Campbell, and Thomas [94], we only know the average %filled and time per event values, which have been taken from their paper. Their algorithms were implemented in Python and tested on a computing cluster equipped with a combination of 2.6 GHz and 2.9 GHz processors running CentOS 6.3. It is not possible to evaluate the real speed of the computers used in their experiments, because there is no detail on the brand and type of processors used and because of the combination of processors running at two different speeds. We notice that their configuration appears to similar to the computer we adopted for our tests, at least in terms of computing frequency (we recall that our computer runs at 2.667 GHz), but their code is probably slower than ours because they use Python whereas we use C++.

The Offline algorithm shows that the best possible %filled value achievable (with our routing optimize function) is 86.26%. Among the Reoptimization methods with

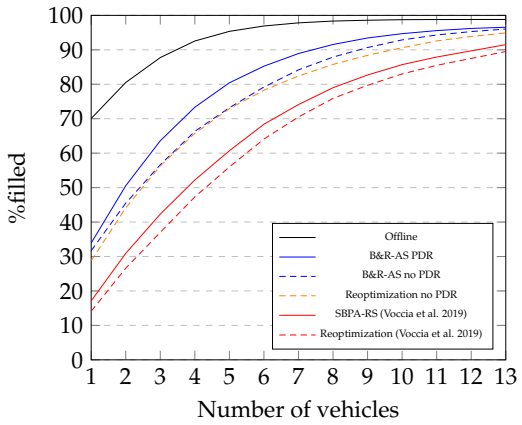
no PDR, our algorithm is faster than the one by Voccia, Campbell, and Thomas [94] and can serve many more requests (54.79% vs. 36.81%). This can be imputed to the efficiency of our routing optimize function. For what concerns SBPA-RS, our implementation is more effective than the one by Voccia, Campbell, and Thomas [94], as it can serve 58.79% of the requests whereas they can serve only 41.99% of the requests. We can also conclude that it is faster, as it takes 0.5 vs. 95.2 seconds of computing time per event, although, as previously discussed, a precise evaluation of the difference in the computer speeds is impossible. A slight reduction in %filled is obtained by the SBPA with the new AS consensus function. However, a decrease in traveled distance and number of routes is also observed. The best results are obtained by B&R-AS. When PDRs are not allowed, B&R-AS can serve 60.46% of the requests, which is about 1.7 percentage points better than our SBPA-RS and 5.7 percentage points better than Reoptimization. When PDRs are allowed, the %filled increases to 61.4%, which is an improvement of 0.9 percentage points. This is due to the larger need for optimization when the problem is more constrained and the vehicles are a scarce resource. The number of events faced by the B&R method with no PDR is about a half of those faced by the SBPA ones. However, their times are larger (about 1.2 seconds per event vs. 0.5) because of the increased complexity of the procedures. Nevertheless, the computing times per event remain very low and this is very important from a managerial point of view, because, no matter the choice adopted for the PDRs, the B&R-AS can be easily used in practice, requiring the decision maker to wait for about one second, which is a negligible time. We can thus conclude that B&R-AS is a viable and effective method for the SDDP.

Table 4.7: Comparison with Voccia, Campbell, and Thomas [94] (three vehicles)

algorithm		%filled	dist.	time	routes	events	time/events
Reoptimization (Voccia et al. 2019)	no PDR	36.81	-	-	-	-	3.3
SBPA-RS (Voccia et al. 2019)	no PDR	41.99	-	-	-	-	95.2
Reoptimization	no PDR	54.79	1218.5	0.2	12.4	87.0	0.0
SBPA-AS	no PDR	58.57	1312.2	106.7	14.1	188.8	0.5
SBPA-RS	no PDR	58.79	1365.2	98.1	15.4	177.0	0.5
B&R-AS	no PDR	60.46	1283.1	97.6	12.4	83.9	1.2
B&R-AS	PDR	61.40	1291.2	126.1	12.9	146.8	0.9
Offline	-	86.26	1284.3	0.4	15.1	1.0	0.4

The positive results that we obtained for the case with three vehicles can also be confirmed for other fleet sizes, as graphically depicted in Figure 4.3. To replicate the same test performed by Voccia, Campbell, and Thomas [94], we executed all algorithms given in the legend of the figure on the same subset of 450 Hom2 instances selected by them, by varying the number of vehicles as shown on the x-axis. Each point in the figure thus corresponds to an average over 450 tests. In the left graph in the figure, the y-axis shows the average %filled value achieved on the different runs. Apart from the Offline algorithm, the best performance is achieved once more by B&R-AS with PDR. Below this method, we can find, in order, B&R-AS with no PDR, Reoptimization and then the algorithms by Voccia, Campbell, and Thomas [94]. Considering the B&R algorithms, the time per event is always larger for B&R-AS with PDR, which confirms the finding in Section 4.5.2 that PDRs slightly increase the computational effort. Both algorithms operate in a range between 1 and 9 seconds per event and scale very well when the number of vehicles increases. In any case, the B&R-AS remains fully compatible with a real use in practice.

Figure 4.3: Percentage of requests filled for Hom2 instances (96 expected requests)



4.5.4 Computational results on large-scale instances

In Figure 4.4, we present the computational evaluation performed to analyze how the B&R behaves on large size instances. We focus on two further sets proposed by Voccia, Campbell, and Thomas [94]: 135 instances Hom3 (on the left part of the figure), where the expected number of requests is equal to 144 (arrival rate equal to 0.003 per minute for 100 customer locations, and a time horizon of 480 minutes); and 135 instances Hom4 (on the right) having an expected number of requests equal to 192 (arrival rate increased to 0.004). Voccia, Campbell, and Thomas [94] solved instances with up to 13 vehicles, whereas we rise this value to 18 vehicles for Hom3 and 26 for Hom4. We tested B&R-AS with and without PDR, and compared them against Offline. In the Hom3 graph, apart from Offline, the best %filled values are obtained by B&R-AS. Considering the seconds per event, the B&R scales well when the number of vehicles increases. The required effort is always in the range between 4 and 12 seconds per event. Both algorithms reach the maximum value with 9 vehicles, respectively 12 seconds per event with PDR and 10 seconds per event with no PDR. In the Hom4 graph, once again B&R-AS with PDR achieves the best results. The computational effort scales well despite the larger number of requests and vehicles, and no algorithm exceeds 27 seconds per event with any vehicle tested. We can conclude that even on these larger instances the B&R algorithm is suitable for a pratical use.

4.5.5 Solution structure analysis

We conclude our computational study by analyzing how the solution structure is affected by the time window type and by the choice of the algorithm. We selected a generic Hom2 instance and solved it with Offline, Reoptimization, and B&R-AS no PDR. We considered a fleet of 10 vehicles, and evaluated how many of these vehicles remained waiting at the depot, at each moment of the time period. The outcome of this study is reported in Figure 4.5.

Figure 4.4: Percentage of requests filled for Hom3 instances (144 expected requests) and Hom4 instances (192 expected requests)

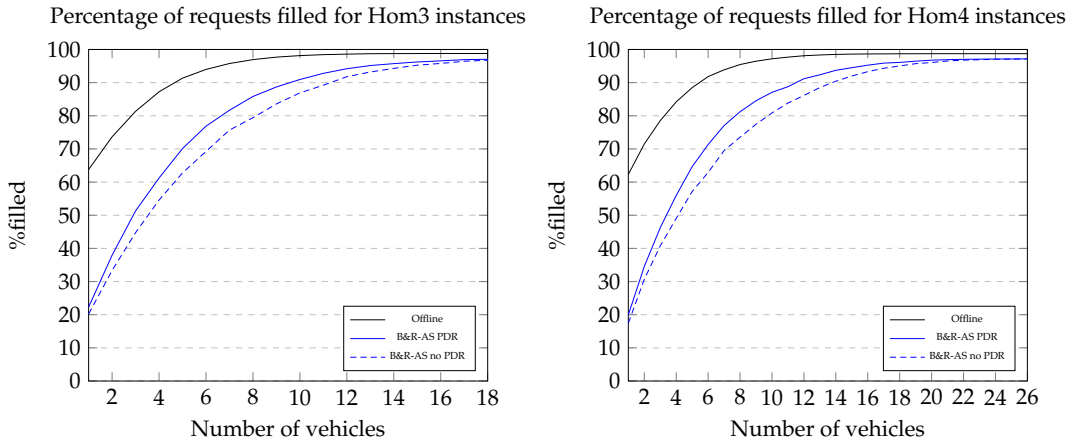
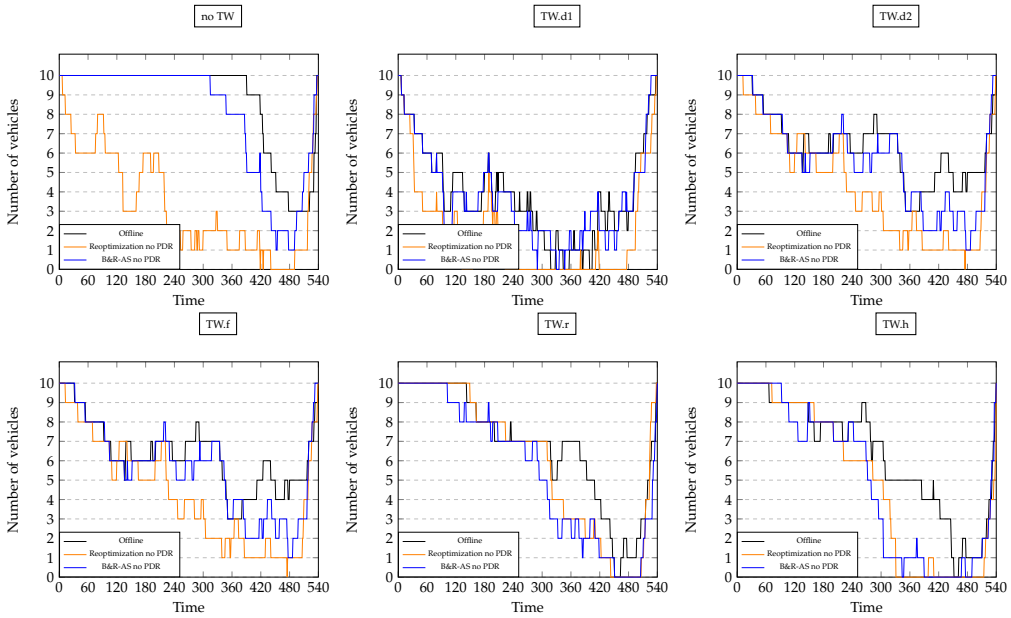


Figure 4.5: Number of vehicles waiting at the depot for a typical instance



For no TW, Offline keeps waiting at the depot all the 10 vehicles as long as possible, and then, around minute 420, starts the routes. This is comprehensible, due to the unrealistic knowledge that this algorithm has of the future. On the contrary, Reoptimization starts the routes very soon, and it never happens throughout the entire time horizon, apart from the beginning and the end, that all 10 vehicles wait at the depot. The behavior of B&R-AS is much more similar to the one of Offline. Indeed, it keeps all the 10 vehicles waiting at the depot until approximately minute 330, after which it starts the routes. Notably, despite the routes started later, the vehicles return earlier at the depot with respect to Reoptimization, and this can be noticed by the fact that there are more vehicles waiting for B&R-AS after minute 450. For TW.d1, TW.d2, and TW.f, all algorithms start the vehicle routes quite early, to be able to satisfy the strict time-window constraints. In this case too, however, we can notice

that Offline tends to keep more vehicles waiting at the depot than B&R-AS, and that B&R-AS, in turn, keeps more vehicles waiting than Reoptimization. This behavior can be noticed all throughout the time horizon, with just some rare exceptions. For TW.r and TW.h, all algorithms keep a high number of vehicles at the depot at the beginning, reducing it quickly after 120 minutes, and then using the vehicles until almost the end of the time horizon.

We performed the same analysis also for B&R-AS PDR, but we obtained similar results. Thus, we opted to not include the results in the figure, because the graphs would be difficult to visualize. In a few words, the results that we obtained indicate that the number of vehicles waiting at the depot for B&R-AS with or without PDR is similar when 10 vehicles are available. When fewer vehicles are available, we could notice that sometimes B&R-AS PDR produced slightly more vehicles waiting at the depot, and sometimes slightly less.

4.6 Conclusions and Future Research

In this paper, we studied dynamic vehicle routing problems where stochastic customers request deliveries with strict and close time windows, and the aim is to maximize served requests and minimize traveled distances. This type of problem is known in the literature as the same-day delivery problem and is of great relevance because it models a number of real-world applications, including the delivery of online purchases. To solve the problem we developed a tailored branch-and-regret algorithm in which sampled scenarios are used to anticipate decisions. We tested the algorithm on a large set of benchmark instances from the literature, obtaining very favorable comparisons. Notably, on a large benchmark set of 4050 instances from Voccia, Campbell, and Thomas [94], the branch-and-regret raises the rate of served requests from about 42% to more than 61% and, at the same time, requires a much shorter computing effort, decreasing the computing time per event from about 95 seconds to just 0.9. The algorithm also scales very well when considering instances with a larger number of vehicles, where it still requires a limited computational effort to produce high-quality solutions.

These good results have been obtained by employing algorithmic features from the literature as well as new techniques, and by performing a careful calibration of the parameter settings. Overall, we found out that it is still important to devote a good effort to optimize the vehicle routes, for which we found convenient to adopt an algorithm based on the adaptive large neighborhood search by Ropke and Pisinger [79]. In addition, it is important to make use of stochastic information. To this aim, as suggested by Bent and Van Hentenryck [11], we made use of consensus functions to select the best set of routes when both real and sample requests are taken into account. We found out that there is still relevant research to be done in the field of consensus functions, but a new function that we introduced proved to lead to better results than the one usually adopted in the literature.

Naive implementations of branch-and-regret algorithms may be very time consuming, as a large number of alternatives must be taken into considerations and optimized. We found out that a good management of the events might consistently decrease the computing time and, at the same time, still allow to get very high rates of served requests. We obtain this positive effect by developing a new way to generate scenarios (correlated-data sampling) and a new branching scheme.

A number of interesting managerial insights may be devised from our study. A company should first steer customers towards wide delivery time windows and

find the appropriate vehicle size according to the expected number of requests. Once this is done, the use of an optimization algorithm such as B&R-AS is highly recommended, as this can lead to an important increase in the service rate at the expense of a negligible waiting time (about one second) for the decision maker. In addition, under certain circumstances such as dispersed time window start times and limited fleet size, a company should opt to equip the drivers with fast communication systems so as to preemptively reroute them back to the depot when needed.

The relationship between preemptive depot returns, waiting strategies, and fleet occupancy is indeed very interesting. When waiting is not imposed, as in the plain reoptimization algorithms, then PDR brings important savings. If, instead, waiting is allowed, as in SBPA and B&R algorithms, then PDR is still important but with less advantages. This can be motivated by the fact that, after waiting, the vehicles depart with a satisfactory number of requests and that these requests are closer to their deadlines, so there is less room for further optimization. This is also impacted by fleet occupancy. If the fleet is composed by a small number of vehicles compared to the number of requests, then waiting becomes unnecessary (as the best strategy is to leave as soon as possible), but PDR is relevant (0.9% more requests filled by B&R-AS). When, instead, the fleet is large, it makes sense to wait at the depot to look for optimized routes. Studying the relationship between waiting strategies, preemptive depot returns, and fleet occupancy appears to be an interesting research direction, even in other related dynamic vehicle routing problems.

There are several interesting future research directions to follow. In terms of methodology, we believe that there is still room for improvements in branch-and-regret algorithms by developing alternative consensus functions, and new mechanisms that make better use of the information from the scenarios. In addition, we believe good results could be obtained by a deep study of immediate request acceptance policies, as in Klapp, Erera, and Toriello [55] and Ulmer and Thomas [90], so as to assign as soon as possible a request to a third-party logistic operator. This could decrease waiting times for the customers, but at the possible expense of an increase in the overall delivery costs.

In terms of optimization problems, as our algorithms are already equipped to solve dynamic pickup and delivery problems, it would be interesting to study their performance on different emerging problem variants. Among these, we would like to cite one-to-many-to-one problems, as in Bruck and Iori [15], where, in addition to the delivery of merchandise, one has to collect further merchandise to be brought back to the depot. This case could involve stochastic customers, stochastic demands and capacitated vehicles. Multi-pickup and delivery problems with time windows (see, e.g., Naccache, Côté, and Coelho [70] and Aziez, Côté, and Coelho [9]) too represent an emerging variant with relevant applications. In these problems, a request is composed of several pickups of different items, followed by a single delivery at the customer location. Stochastic aspects might hence concern both customer and pickup locations. Finally, we mention the class of meal delivery problems (see, e.g., Ulmer et al. [91]), where the customers require food from restaurants and the aim is to deliver it promptly by considering the time in which it will be ready.

Acknowledgments

Jean-François Côté acknowledges support by the Canadian Natural Sciences and Engineering Research Council (NSERC) under grant 2021-04037. Thiago Alves de

Queiroz acknowledges support by the National Council for Scientific and Technological Development (CNPq - grants 405369/2021-2 and 311185/2020-7) and the State of Goiás Research Foundation (FAPEG). Manuel Iori acknowledges support from the University of Modena and Reggio Emilia under grant FAR 2018. We thank Compute Canada for providing high-performance computing facilities. We thank three anonymous reviewers whose comments greatly improved the quality of the paper.

Chapter 5

Conclusions

In this thesis, we focus on problems related to dynamic transportation systems. First, we propose a survey on scheduling Automated Guided Vehicles (AGVs) and show the challenges and future directions in this field. Second, we extend the survey considering different aspects of the coordination of AGVs, also in this case discussing promising future research directions. Third, we present an innovative branch-and-regret algorithm to handle a fleet of vehicles to serve urgent customer requests.

In Chapter 2 of the thesis, we addressed the planning problem of AGVs in complex logistic systems. Our focus was specifically on the scheduling problem of AGVs, which can be modeled as a Pickup and Delivery Problem (PDP), and holds significant importance for E80 Group and numerous real-world contexts. Through the formulation of the PDP and an analysis of real-world applications and existing literature, we successfully identified research challenges and opportunities for future investigations. Our findings provide a valuable guide for researchers involved in AGV system optimization, with the potential to significantly enhance the efficiency of logistic operations. The chapter begins with a literature review on the surveys related to scheduling AGVs, then propose a model for a formal formulation of the problem, and finally lists the future research directions present in the literature.

In Chapter 3, we extend Chapter 2 considering the coordination of a fleet of AGVs within an intralogistic system in general. The aspects addressed include the scheduling of AGVs, battery management, multi-load scenarios, path planning, conflict avoidance, and the integration of scheduling AGVs with other systems. Following a comprehensive review of books and surveys that constitute the current state of the art in AGV systems, we turn our attention to the scheduling problem. We introduce a mathematical model for the PDP incorporating battery management considerations and a variant accommodating multi-load vehicles. Finally, we present a detailed list of papers that highlight current challenges and indicate future research directions found in the literature.

In Chapter 4, we concentrated on the issue of urgent deliveries, known in the literature as the Same-Day Delivery Problem, which has gained increasing importance in the realm of e-commerce services. To tackle this problem, we proposed an innovative branch-and-regret algorithm that utilizes sampled scenarios to anticipate future events, while employing an iterative adaptive large neighborhood search to optimize routing plans. The branch-and-regret algorithm distinguishes itself by effectively handling stochastic requests and achieving outstanding performance in terms of served requests, traveled distance, and computational effort, surpassing recent literature results. Our contributions in modeling the subproblem, generating scenarios, managing consensus functions, and implementing the branching scheme improve the solution to the same-day delivery problem.

From an industrial perspective, the papers collected in the literature review conducted in Chapters 2 and 3 adopt a theoretical and analytical approach, albeit with

limited consideration for real-world applications. Specifically, many articles study specific problems by addressing only a subset of the constraints encountered in industrial scenarios. Consequently, their utility for real-world applications is somewhat constrained. I find it relevant to encounter discussions on dynamic management and unexpected fault requirements as future research directions within the collected papers, as these are crucial aspects for addressing industrial applications effectively.

Chapter 4 introduces a B&R algorithm tailored for real-world applicability. The algorithm demonstrates proficiency in handling up to 192 requests per day and a fleet size of up to 13 vehicles, achieving execution times of approximately 1 second in the tests presented. This exemplifies a scenario typical of small or medium-sized e-commerce enterprises. Future research directions could explore the performance of the algorithm with larger instances and assess the limit within which the algorithm can quickly produce a satisfactory solution, useful for industrial applications.

The application of operations research algorithms to real-world cases holds primary importance for me, as evidenced by my decision to pursue an industrial doctorate. My main research was dedicated to a project within E80 Group, focusing on order scheduling within an AGV system. The objective is to optimize system throughput while minimizing order delays. The algorithm necessitates frequent recomputation, addressing a fleet involving up to 200 AGVs. Operating within a dynamic environment where information changes quickly, the algorithm must manage unexpected system faults. Furthermore, the product aspires to be useful to numerous customer plants, necessitating adaptability to accommodate highly customized layouts. Presently, the algorithm is operational in four real-world plants, with one boasting an AGV fleet exceeding 80 units. However, the non-disclosure policy of the company E80 Group prohibits the publication of detailed information regarding this significant work of my research. Therefore, I refrain from providing further insights into this project.

In conclusion, this doctoral thesis has made significant contributions to the advancement of operations research and optimization in the field of AGV transportation systems and same-day deliveries. Through the survey on AGV scheduling, the survey on the coordination of a fleet of AGVs, and the development of the branch-and-regret algorithm for urgent deliveries, we have provided new perspectives and innovative solutions. The results obtained offer advantages for both companies operating in the logistics sector and consumers benefiting from e-commerce services.

As future research, I plan to develop new innovative optimization algorithms for the scheduling of AGVs in the context of real-world industrial systems, continuing my research and innovation activity at E80 Group.

Bibliography

[1] M. Ali and W. U. Khan. "Implementation issues of AGVs in flexible manufacturing system: A Review". In: *Global Journal of Flexible Systems Management* 11 (2010), pp. 55–62.

[2] C. Archetti, D. Feillet, A. Mor, and M. Speranza. "Dynamic traveling salesman problem with stochastic release dates". In: *European Journal of Operational Research* 280.3 (2020), pp. 832–844.

[3] Y. Arda, Y. Crama, D. Kronus, T. Pironet, and P. Van Hentenryck. "Multi-period vehicle loading with stochastic release dates". In: *EURO Journal on Transportation and Logistics* 3.2 (2014), pp. 93–119.

[4] A. M. Arslan, N. Agatz, L. Kroon, and R. Zuidwijk. "Crowdsourced delivery–a dynamic pickup and delivery problem with ad hoc drivers". In: *Transportation Science* 53.1 (2019), pp. 222–235.

[5] A. Asef-Vaziri and G. Laporte. "Loop based facility planning and material handling". In: *European Journal of Operational Research* 164.1 (2005), pp. 1–11.

[6] N. Azi, M. Gendreau, and J.-Y. Potvin. "An exact algorithm for a vehicle routing problem with time windows and multiple use of vehicles". In: *European Journal of Operational Research* 202.3 (2010), pp. 756–763.

[7] N. Azi, M. Gendreau, and J.-Y. Potvin. "A dynamic vehicle routing problem with multiple delivery routes". In: *Annals of Operations Research* 199.1 (2012), pp. 103–112.

[8] N. Azi, M. Gendreau, and J.-Y. Potvin. "An adaptive large neighborhood search for a vehicle routing problem with multiple routes". In: *Computers & Operations Research* 41 (2014), pp. 167–173.

[9] I. Aziez, J.-F. Côté, and L. C. Coelho. "Exact algorithms for the multi-pickup and delivery problem with time windows". In: *European Journal of Operational Research* 284.3 (2020), pp. 906–919.

[10] M. Battarra, M. Monaci, and D. Vigo. "An adaptive guidance approach for the heuristic solution of a minimum multiple trip vehicle routing problem". In: *Computers & Operations Research* 36 (2009), pp. 3041–3050.

[11] R. W. Bent and P. Van Hentenryck. "Scenario-Based Planning for Partially Dynamic Vehicle Routing with Stochastic Customers". In: *Operations Research* 52.2 (2004), pp. 977–987.

[12] R. W. Bent and P. Van Hentenryck. "Waiting and Relocation Strategies in Online Stochastic Vehicle Routing". In: *Proceedings of the 20th International Joint Conference on Artifical Intelligence*. 2007, pp. 1816–1821.

[13] M. Boccia, A. Masone, C. Sterle, and T. Murino. "The parallel AGV Scheduling Problem with battery constraints: a new formulation and a matheuristic approach". In: *European Journal of Operational Research* (2022, forthcoming).

[14] E. Bordelon Hoff and B. R. Sarker. "An overview of path design and dispatching methods for automated guided vehicles". In: *Integrated Manufacturing Systems* 9.5 (1998), pp. 296–307.

[15] P. B. Bruck and M. Iori. "Non-elementary formulations for single vehicle routing problems with pickups and deliveries". In: *Operations Research* 65.6 (2017), pp. 1597–1614.

[16] S. Bunte and N. Kliewer. "An overview on vehicle scheduling models". In: *Public Transport* 1.4 (2009), pp. 299–317.

[17] D. Cattaruzza, N. Absi, and D. Feillet. "The Multi-Trip Vehicle Routing Problem with Time Windows and Release Dates". In: *Transportation Science* 50.2 (2016), pp. 676–693.

[18] D. Cattaruzza, N. Absi, and D. Feillet. "Vehicle routing problems with multiple trips". In: *Annals of Operations Research* 271 (2018), pp. 127–159.

[19] D. Cattaruzza, N. Absi, D. Feillet, and T. Vidal. "A memetic algorithm for the Multi Trip Vehicle Routing Problem". In: *European Journal of Operational Research* 236.3 (2014), pp. 833–848.

[20] V. Chawla, A. Chanda, S. Angra, and S. Rani. "Simultaneous Dispatching and Scheduling of Multi-Load AGVs in FMS-A Simulation Study". In: *Materials Today: Proceedings* 5.11, Part 3 (2018), pp. 25358–25367.

[21] J. C. Chen, T.-L. Chen, and Y.-C. Teng. "Meta-model based simulation optimization for automated guided vehicle system under different charging mechanisms". In: *Simulation Modelling Practice and Theory* 106 (2021), p. 102208.

[22] X. Chen, S. He, Y. Zhang, L. C. Tong, P. Shang, and X. Zhou. "Yard crane and AGV scheduling in automated container terminal: A multi-robot task allocation framework". In: *Transportation Research Part C: Emerging Technologies* 114 (2020), pp. 241–271.

[23] C. G. Co and J. Tanchoco. "A review of research on AGVS vehicle management". In: *Engineering Costs and Production Economics* 21.1 (1991), pp. 35–42.

[24] Q.-V. Dang, N. Singh, I. Adan, T. Martagan, and D. van de Sande. "Scheduling heterogeneous multi-load AGVs with battery constraints". In: *Computers & Operations Research* 136 (2021), p. 105517.

[25] Q.-V. Dang, N. Singh, I. Adan, T. Martagan, and D. van de Sande. "Scheduling heterogeneous multi-load AGVs with battery constraints". In: *Computers & Operations Research* 136 (2021), p. 105517.

[26] I. Dayarian and M. Savelsbergh. "Crowdshipping and same-day delivery: Employing in-store customers to deliver online orders". In: *Production and Operations Management* 29.9 (2020), pp. 2153–2174.

[27] I. Dayarian, M. Savelsbergh, and J.-P. Clarke. "Same-Day Delivery with Drone Resupply". In: *Transportation Science* 54.1 (2020), pp. 229–249.

[28] M. De Ryck, D. Pissoort, T. Holvoet, and E. Demeester. "Decentral task allocation for industrial AGV-systems with resource constraints". In: *Journal of Manufacturing Systems* 59 (2021), pp. 310–319.

[29] M. De Ryck, M. Versteyhe, and F. Debrouwere. "Automated guided vehicle systems, state-of-the-art control algorithms and techniques". In: *Journal of Manufacturing Systems* 54 (2020), pp. 152–173.

[30] M. De Ryck, M. Versteyhe, and K. Shariatmadar. "Resource management in decentralized industrial Automated Guided Vehicle systems". In: *Journal of Manufacturing Systems* 54 (2020), pp. 204–214.

[31] A. Djenadi and B. Mendil. "Energy-aware task allocation strategy for multi robot system". In: *International Journal of Modelling and Simulation* 42.1 (2022), pp. 153–167.

[32] *E80 Group*. URL: https://www.e80group.com/en/ (visited on 07/2023).

[33] H. Fazlollahtabar and M. Saidi-Mehrabad. "Methodologies to optimize automated guided vehicle scheduling and routing problems: a review study". In: *Journal of Intelligent & Robotic Systems* 77.3 (2015), pp. 525–545.

[34] B. Fleischmann. "The vehicle routing problem with multiple use of vehicles". PhD thesis. Fachbereich Wirtschaftswissenschaften, Universität Hamburg, 1990.

[35] G. Fragapane, R. de Koster, F. Sgarbossa, and J. O. Strandhagen. "Planning and control of autonomous mobile robots for intralogistics: Literature review and research agenda". In: *European Journal of Operational Research* 294.2 (2021), pp. 405–426.

[36] K. Fransen, J. van Eekelen, A. Pogromsky, M. Boon, and I. Adan. "A dynamic path planning approach for dense, large, grid-based automated guided vehicle systems". In: *Computers & Operations Research* 123 (2020), p. 105046.

[37] T. Ganesharajah, N. G. Hall, and C. Sriskandarajah. "Design and operational issues in AGV-served manufacturing systems". In: *Annals of Operations Research* 76.0 (1998), pp. 109–154.

[38] M. Gendreau, F. Guertin, J.-Y. Potvin, and E. Taillard. "Parallel tabu search for real-time vehicle routing and dispatching". In: *Transportation Science* 33.4 (1999), pp. 381–390.

[39] G. Ghiani, E. Manni, A. Quaranta, and C. Triki. "Anticipatory algorithms for same-day courier dispatching". In: *Transportation Research Part E: Logistics and Transportation Review* 45.1 (2009), pp. 96–106.

[40] M. Godinho Filho, C. F. Barco, and R. F. Tavares Neto. "Using Genetic Algorithms to solve scheduling problems on flexible manufacturing systems (FMS): a literature survey, classification and analysis". In: *Flexible Services and Manufacturing Journal* 26 (2014), pp. 408–431.

[41] W. K. K. Haneveld and M. H. van der Vlerk. "Stochastic integer programming: General models and algorithms". In: *Annals of Operations Research* 85 (1999), pp. 39–57.

[42] Z. He, V. Aggarwal, and S. Y. Nof. "Differentiated service policy in smart warehouse automation". In: *International Journal of Production Research* 56.22 (2018), pp. 6956–6970.

[43] W. J. van Heeswijk, M. R. Mes, and J. M. Schutten. "The delivery dispatching problem with time windows for urban consolidation centers". In: *Transportation science* 53.1 (2019), pp. 203–221.

[44] J. Heger and T. Voß. "Dynamic priority based dispatching of AGVs in flexible job shops". In: *Procedia CIRP* 79 (2019). 12th CIRP Conference on Intelligent Computation in Manufacturing Engineering, 18-20 July 2018, Gulf of Naples, Italy, pp. 445–449.

[45] Y. Hu, H. Yang, and Y. Huang. "Conflict-free scheduling of large-scale multi-load AGVs in material transportation network". In: *Transportation Research Part E: Logistics and Transportation Review* 158 (2022), p. 102623.

[46] L. M. Hvattum, A. Løkketangen, and G. Laporte. "Solving a Dynamic and Stochastic Vehicle Routing Problem with a Sample Scenario Hedging Heuristic". In: *Transportation Science* 40.4 (2006), pp. 421–438.

[47] L. M. Hvattum, A. Løkketangen, and G. Laporte. "A branch-and-regret heuristic for stochastic and dynamic vehicle routing problems". In: *Networks: An International Journal* 49.4 (2007), pp. 330–340.

[48] D. Ivanov, C. S. Tang, A. Dolgui, D. Battini, and A. Das. "Researchers' perspectives on Industry 4.0: multi-disciplinary analysis and opportunities for operations management". In: *International Journal of Production Research* 59.7 (2021), pp. 2055–2078.

[49] Q. S. Kabir and Y. Suzuki. "Increasing manufacturing flexibility through battery management of automated guided vehicles". In: *Computers & Industrial Engineering* 117 (2018), pp. 225–236.

[50] Q. S. Kabir and Y. Suzuki. "Comparative analysis of different routing heuristics for the battery management of automated guided vehicles". In: *International Journal of Production Research* 57.2 (2019), pp. 624–641.

[51] E. Kaoud, M. A. El-Sharief, and M. El-Sebaie. "Scheduling problems of automated guided vehicles in job shop, flow shop, and container terminals". In: *2017 4th International Conference on Industrial Engineering and Applications (ICIEA)*. 2017, pp. 60–65.

[52] R. E. King and C. Wilson. "A review of automated guided-vehicle systems design and scheduling". In: *Production Planning & Control* 2.1 (1991), pp. 44–51.

[53] M. A. Klapp, A. L. Erera, and A. Toriello. "The Dynamic Dispatch Waves Problem for same-day delivery". In: *European Journal of Operational Research* 271.2 (2018), pp. 519–534.

[54] M. A. Klapp, A. L. Erera, and A. Toriello. "The One-Dimensional Dynamic Dispatch Waves Problem". In: *Transportation Science* 52.2 (2018), pp. 402–415.

[55] M. A. Klapp, A. L. Erera, and A. Toriello. "Request acceptance in same-day delivery". In: *Transportation Research Part E: Logistics and Transportation Review* 143 (2020), p. 102083.

[56] T. Le-Anh and M. De Koster. "A review of design and control of automated guided vehicle systems". In: *European Journal of Operational Research* 171.1 (2006), pp. 1–23.

[57] C.-Y. Lee, L. Lei, and M. Pinedo. "Current trends in deterministic scheduling". In: *Annals of Operations Research* 70.0 (1997), pp. 1–41.

[58] V. I. Levenshtein. "Binary codes capable of correcting deletions, insertions, and reversals". In: *Soviet Physics Doklady* 10.8 (1966), pp. 707–710.

[59] I. I. Lin and H. S. Mahmassani. "Can online grocers deliver? Some logistics considerations". In: *Transportation Research Record* 1817 (2002), pp. 17–24.

[60] Y. Lin, Y. Xu, J. Zhu, X. Wang, L. Wang, and G. Hu. "MLATSO: A method for task scheduling optimization in multi-load AGVs-based systems". In: *Robotics and Computer-Integrated Manufacturing* 79 (2023), p. 102397.

[61] D. Liu, P. Yan, Z. Pu, Y. Wang, and E. I. Kaisar. "Hybrid artificial immune algorithm for optimizing a Van-Robot E-grocery delivery system". In: *Transportation Research Part E: Logistics and Transportation Review* 154 (2021), p. 102466.

[62] A. Løkketangen and D. L. Woodruff. "Progressive hedging and tabu search applied to mixed integer (0,1) multi-stage stochastic programming". In: *Journal of Heuristics* 2 (1996), pp. 111–128.

[63] W. Lu, S. Guo, T. Song, and Y. Li. "Analysis of multi-AGVs management system and key issues: A review". In: *Computer Modeling in Engineering & Sciences* 131.3 (2022), pp. 1197–1227.

[64] X. Lyu, Y. Song, C. He, Q. Lei, and W. Guo. "Approach to Integrated Scheduling Problems Considering Optimal Number of Automated Guided Vehicles and Conflict-Free Routing in Flexible Manufacturing Systems". In: *IEEE Access* 7 (2019).

[65] N. Ma, C. Zhou, and A. Stephen. "Simulation model and performance evaluation of battery-powered AGV systems in automated container terminals". In: *Simulation Modelling Practice and Theory* 106 (2021), p. 102146.

[66] E. Manafi, R. Tavakkoli-Moghaddam, and M. Mahmoodjanloo. "A centroid opposition-based coral reefs algorithm for solving an automated guided vehicle routing problem with a recharging constraint". In: *Applied Soft Computing* 128 (2022), p. 109504.

[67] A. Mingozzi, R. Roberti, and P. Toth. "An Exact Algorithm for the Multitrip Vehicle Routing Problem". In: *INFORMS Journal on Computing* 25.2 (2013), pp. 193–207.

[68] A. S. Minkoff. "A Markov Decision Model and Decomposition Heuristic for Dynamic Vehicle Dispatching". In: *Operations Research* 41.1 (1993), pp. 77–90.

[69] S. Mitrović-Minić and G. Laporte. "Waiting strategies for the dynamic pickup and delivery problem with time windows". In: *Transportation Research Part B: Methodological* 38.7 (2004), pp. 635–655.

[70] S. Naccache, J.-F. Côté, and L. C. Coelho. "The multi-pickup and delivery problem with time windows". In: *European Journal of Operational Research* 269.1 (2018), pp. 353–362.

[71] D. Naeem, M. Gheith, and A. Eltawil. "A comprehensive review and directions for future research on the integrated scheduling of quay cranes and automated guided vehicles and yard cranes in automated container terminals". In: *Computers & Industrial Engineering* 179 (2023), p. 109149.

[72] R. Paradiso, R. Roberti, D. Laganà, and W. Dullaert. "An Exact Solution Framework for Multitrip Vehicle-Routing Problems with Time Windows". In: *Operations Research* 66.1 (2020), pp. 180–198.

[73] V. Pillac, M. Gendreau, C. Guéret, and A. L. Medaglia. "A review of dynamic vehicle routing problems". In: *European Journal of Operational Research* 225.1 (2013), pp. 1–11.

[74] J.-Y. Potvin and J.-M. Rousseau. "A parallel route building algorithm for the vehicle routing and scheduling problem with time windows". In: *European Journal of Operational Research* 66.3 (1993), pp. 331–340.

[75] L. Qiu, W.-J. Hsu, S.-Y. Huang, and H. Wang. "Scheduling and routing algorithms for AGVs: a survey". In: *International Journal of Production Research* 40.3 (2002), pp. 745–760.

[76] H. F. Rahman and I. Nielsen. "Scheduling automated transport vehicles for material distribution systems". In: *Applied Soft Computing* 82 (2019), p. 105552.

[77] H. Rashidi, F. Matinfar, and F. Parand. "Automated guided vehicles-a review on applications, problem modeling and solutions". In: *International Journal of Transportation Engineering* 8.3 (2021), pp. 261–278.

[78] U. Ritzinger, J. Puchinger, and R. F. Hartl. "A survey on dynamic and stochastic vehicle routing problems". In: *International Journal of Production Research* 54.1 (2016), pp. 215–231.

[79] S. Ropke and D. Pisinger. "An Adaptive Large Neighborhood Search Heuristic for the Pickup and Delivery Problem with Time Windows". In: *Transportation Science* 40.4 (2006), pp. 455–472.

[80] T. Schmidt, K.-B. Reith, N. Klein, and M. Däumler. "Research on Decentralized Control Strategies for Automated Vehicle-based In-house Transport Systems– a Survey". In: *Logistics Research* 13.1 (2020).

[81] N. Singh, Q.-V. Dang, A. Akcay, I. Adan, and T. Martagan. "A matheuristic for AGV scheduling with battery constraints". In: *European Journal of Operational Research* 298.3 (2022), pp. 855–873.

[82] Y. Song, M. W. Ulmer, B. W. Thomas, and W. W. Stein. "Building Trust in Home Services – Stochastic Team-Orienteering with Consistency Constraints". In: *Transportation Science* 54.3 (2020), pp. 823–853.

[83] R. Stahlbock and S. Voß. "Vehicle Routing Problems and Container Terminal Operations – An Update of Research". In: *The Vehicle Routing Problem: Latest Advances and New Challenges*. Ed. by B. Golden, S. Raghavan, and E. Wasil. Boston, MA: Springer US, 2008, pp. 551–589.

[84] P. Sun, L. P. Veelenturf, M. Hewitt, and T. Van Woensel. "Adaptive large neighborhood search for the time-dependent profitable pickup and delivery problem with time windows". In: *Transportation Research Part E: Logistics and Transportation Review* 138 (2020), p. 101942.

[85] P. Z. Sun, J. You, S. Qiu, E. Q. Wu, P. Xiong, A. Song, H. Zhang, and T. Lu. "AGV-Based Vehicle Transportation in Automated Container Terminals: A Survey". In: *IEEE Transactions on Intelligent Transportation Systems* 24.1 (2023), pp. 341–356.

[86] G. Tang, C. Tang, C. Claramunt, X. Hu, and P. Zhou. "Geometric A-Star Algorithm: An Improved A-Star Algorithm for AGV Path Planning in a Port Environment". In: *IEEE Access* 9 (2021), pp. 59196–59210.

[87] G. Tirado, L. M. Hvattum, K. Fagerholt, and J.-F. Cordeau. "Heuristics for dynamic and stochastic routing in industrial shipping". In: *Computers & Operations Research* 40.1 (2013), pp. 253–263.

[88] M. W. Ulmer. "Dynamic Pricing and Routing for Same-Day Delivery". In: *Transportation Science* 54.4 (2020), pp. 855–1152.

[89] M. W. Ulmer, J. C. Goodson, D. C. Mattfeld, and B. W. Thomas. "On Modeling Stochastic Dynamic Vehicle Routing Problems". In: *EURO Journal on Transportation and Logistics* 9.2 (2020), p. 100008.

[90] M. W. Ulmer and B. W. Thomas. "Meso-parametric value function approximation for dynamic customer acceptances in delivery routing". In: *European Journal of Operational Research* 285 (2020), pp. 183–195.

[91] M. W. Ulmer, B. W. Thomas, A. M. Campbell, and N. Woyak. "The Restaurant Meal Delivery Problem: Dynamic Pickup and Delivery with Deadlines and Random Ready Times". In: *Transportation Science* 55.1 (2021), pp. 75–100.

[92] M. W. Ulmer, B. W. Thomas, and D. C. Mattfeld. "Preemptive depot returns for dynamic same-day delivery". In: *EURO Journal on Transportation and Logistics* 8.4 (2019), pp. 327–361.

[93] I. F. Vis. "Survey of research in the design and control of automated guided vehicle systems". In: *European Journal of Operational Research* 170.3 (2006), pp. 677–709.

[94] S. A. Voccia, A. M. Campbell, and B. W. Thomas. "The Same-Day Delivery Problem for Online Purchases". In: *Transportation Science* 53.1 (2019), pp. 167–184.

[95] C. Xie and T. T. Allen. "Simulation and experimental design methods for job shop scheduling with material handling: a survey". In: *The International Journal of Advanced Manufacturing Technology* 80.1-4 (2015), pp. 233–243.

[96] R. Yan, L. Jackson, and S. Dunnett. "A study for further exploring the advantages of using multi-load automated guided vehicles". In: *Journal of Manufacturing Systems* 57 (2020), pp. 19–30.

[97] Z. Zhang, J. Chen, and Q. Guo. "Application of Automated Guided Vehicles in Smart Automated Warehouse Systems: A Survey". In: *Computer Modeling in Engineering & Sciences* 134.3 (2023).

[98] M. Zhong, Y. Yang, Y. Dessouky, and O. Postolache. "Multi-AGV scheduling for conflict-free path planning in automated container terminals". In: *Computers & Industrial Engineering* 142 (2020), p. 106371.

[99] W.-Q. Zou, Q.-K. Pan, and M. F. Tasgetiren. "An effective iterated greedy algorithm for solving a multi-compartment AGV scheduling problem in a matrix manufacturing workshop". In: *Applied Soft Computing* 99 (2021), p. 106945.

Appendices

Appendix A

List of Acronyms

A.1 Acronyms, definitions

3PL Third-Party Logistic Operator.

ACT Automated Container Terminals.

ADP Approximate Dynamic Programming.

AGV Automated Guided Vehicle.

ALNS Adaptive Large Neighborhood Search.

AMR Autonomous Mobile Robot.

AP Assignment Problem.

AS Assignment Similarity.

B&R Branch-and-Regret.

CDS Correlated-Data Sampling.

COCRO Centroid Opposition-based Coral Reefs Optimization.

CT Container Terminals.

DPDP Pickup and Delivery Problem with Time Windows and Release Dates.

ED Edit Distance.

FMS Flexible Manufacturing Systems.

MDP Markov Decision Process.

MTTOPTW Multi-Trip Team Orienteering Problem with Time Windows.

MTVRP Multi-Trip Vehicle Routing Problem.

NRRP National Recovery and Resilience Plan.

PDP Pickup and Delivery Problem.

PDR Preemptive Depot Return.

PHH Progressive Hedging Heuristic.

R&D Research and Development.

RS Route Similarity.

SBPA Scenario-based Planning Approach.

SDDP Same-Day Delivery Problem.

SDVRP Stochastic Dynamic Vehicle Routing Problem.

VRP Vehicle Routing Problem.