Control systems and optimization methods for industrial logistics

Department of Electrical and Information Engineering
ELECTRICAL AND INFORMATION ENGINEERING
Ph.D. Program
SSD: ING-INF/04– SYSTEMS AND CONTROL ENGINEERING

Final Dissertation

Control Systems and Optimization Methods for Industrial Logistics

---

by
Giulia Tresca

Supervisors:

Prof. Engr. Mariagrazia Dotoli

Dr. Engr. Graziana Cavone

Engr. Antonio Cerviotti

Coordinator of Ph.D. Program:
Prof. Engr. Mario Carpentieri

Course n°36, 01/11/2020-31/10/2023

Politecnico di Bari

LIBERATORIA PER L'ARCHIVIAZIONE DELLA TESI DI DOTTORATO

Al Magnifico Rettore
del Politecnico di Bari

La sottoscritta Tresca Giulia nata il 01/05/1996 a Barletta (BAT) residente in via Sant'Antonio n° 60 a Barletta (BAT) C.a.p. 76121 - e-mail personale giuliatresca01@gmail.com, iscritta al 3° anno di Corso di Dottorato di Ricerca in Ingegneria Elettrica e dell'Informazione, ciclo XXXVI ed essendo stato ammesso a sostenere l'esame finale con la prevista discussione della tesi dal titolo:

Control Systems and Optimization Methods for Industrial Logistics

DICHIARA

1) di essere consapevole che, ai sensi del D.P.R. n. 445 del 28.12.2000, le dichiarazioni mendaci, la falsità negli atti e l'uso di atti falsi sono puniti ai sensi del codice penale e delle Leggi speciali in materia, e che nel caso ricorressero dette ipotesi, decade fin dall'inizio e senza necessità di nessuna formalità dai benefici conseguenti al provvedimento emanato sulla base di tali dichiarazioni;
2) di essere iscritto al Corso di Dottorato Ricerca in Ingegneria Elettrica e dell'Informazione, ciclo XXXVI, corso attivato ai sensi del "*Regolamento dei Corsi di Dottorato di ricerca del Politecnico di Bari"*, emanato con D.R. n.286 del 01.07.2013;
3) di essere pienamente a conoscenza delle disposizioni contenute nel predetto Regolamento in merito alla procedura di deposito, pubblicazione e autoarchiviazione della tesi di dottorato nell'Archivio Istituzionale ad accesso aperto alla letteratura scientifica;
4) di essere consapevole che attraverso l'autoarchiviazione delle tesi nell'Archivio Istituzionale ad accesso aperto alla letteratura scientifica del Politecnico di Bari (IRIS-POLIBA), l'Ateneo archivierà e renderà consultabile in rete (nel rispetto della Policy di Ateneo di cui al D.R. 642 del 13.11.2015) il testo completo della tesi di dottorato, fatta salva la possibilità di sottoscrizione di apposite licenze per le relative condizioni di utilizzo (di cui al sito http://www.creativecommons.it/Licenze), e fatte salve, altresì, le eventuali esigenze di "embargo", legate a strette considerazioni sulla tutelabilità e sfruttamento industriale/commerciale dei contenuti della tesi, da rappresentarsi mediante compilazione e sottoscrizione del modulo in calce (Richiesta di embargo);
5) che la tesi da depositare in IRIS-POLIBA, in formato digitale (PDF/A) sarà del tutto identica a quelle **consegnate**/inviate/da inviarsi ai componenti della commissione per l'esame finale e a qualsiasi altra copia depositata presso gli Uffici del Politecnico di Bari in forma cartacea o digitale, ovvero a quella da discutere in sede di esame finale, a quella da depositare, a cura dell'Ateneo, presso le Biblioteche Nazionali Centrali di Roma e Firenze e presso tutti gli Uffici competenti per legge al momento del deposito stesso, e che di conseguenza va esclusa qualsiasi responsabilità del Politecnico di Bari per quanto riguarda eventuali errori, imprecisioni o omissioni nei contenuti della tesi;
6) che il contenuto e l'organizzazione della tesi è opera originale realizzata dal sottoscritto e non compromette in alcun modo i diritti di terzi, ivi compresi quelli relativi alla sicurezza dei dati personali; che pertanto il Politecnico di Bari ed i suoi funzionari sono in ogni caso esenti da responsabilità di qualsivoglia natura: civile, amministrativa e penale e saranno dal sottoscritto tenuti indenni da qualsiasi richiesta o rivendicazione da parte di terzi;
7) che il contenuto della tesi non infrange in alcun modo il diritto d'Autore né gli obblighi connessi alla salvaguardia di diritti morali od economici di altri autori o di altri aventi diritto, sia per testi, immagini, foto, tabelle, o altre parti di cui la tesi è composta.

Bari 18/12/2023 Firma 

La sottoscritta, con l'autoarchiviazione della propria tesi di dottorato nell'Archivio Istituzionale ad accesso aperto del Politecnico di Bari (POLIBA-IRIS), pur mantenendo su di essa tutti i diritti d'autore, morali ed economici, ai sensi della normativa vigente (Legge 633/1941 e ss.mm.ii.),

CONCEDE

- al Politecnico di Bari il permesso di trasferire l'opera su qualsiasi supporto e di convertirla in qualsiasi formato al fine di una corretta conservazione nel tempo. Il Politecnico di Bari garantisce che non verrà effettuata alcuna modifica al contenuto e alla struttura dell'opera.
- al Politecnico di Bari la possibilità di riprodurre l'opera in più di una copia per fini di sicurezza, back-up e conservazione.

Bari 18/12/2023 Firma 

Politecnico di Bari

Politecnico di Bari

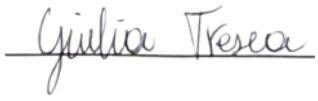
Department of Electrical and Information Engineering
ELECTRICAL AND INFORMATION ENGINEERING
Ph.D. Program
SSD: ING-INF/04– SYSTEMS AND CONTROL ENGINEERING

Final Dissertation

Control Systems and Optimization Methods for Industrial Logistics

---

by
Giulia Tresca:



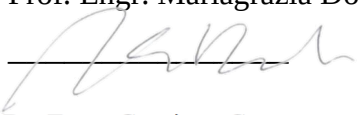
Referees:

Prof. Engr. Carla Seatzu

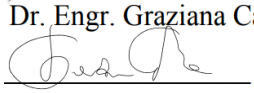
Prof. Engr. Silvia Siri

Supervisors:

Prof. Engr. Mariagrazia Dotoli



Dr. Engr. Graziana Cavone



Engr. Antonio Cerviotti

Coordinator of Ph.D Program:

Prof. Engr. Mario Carpentieri

Course n°36, 01/11/2020-31/10/2023

Abstract

Industry 4.0, is the well-known term used to represent the fourth industrial revolution, characterized by the integration of digital technologies, automation, and data-driven decision-making into manufacturing and industrial processes. An important aspect, on which both researchers and companies are investigating, is the practical implementation of Industry 4.0 in logistics, considering case studies and best practices from leading companies. As a matter of fact, this doctoral thesis improve some of the most time-consuming and expansive operations in logistics by using the most promising control and optimization techniques. The thesis begins by providing a comprehensive overview of the Industry 4.0 framework and discusses the potential benefits of adopting Industry 4.0 principles (i.e., improved efficiency, reduced costs, and enhanced flexibility) in the logistic sector. Then, after the analysis of the main classification used for the logistic applications, the work is developed following two different functional contexts, that are the internal and the external logistics, applied to several application operational scenarios. In particular, for the first part, the two main problems tackled are the ones related to automated storage systems (i.e., systems designed to automate the process of storing and retrieving goods or materials within a warehouse or storage environment) that are the development of the automated bin packing algorithm and the optimization of the design of a particular kind of automated warehouse named Vertical Lift Module warehouse. On the other hand, the addressed in the second part are related to the management of the delivery planning and online replanning operations: starting with the enhancement of the cargo inside the container (i.e., the container loading problem), to the optimization of the vehicles routing both offline and in real-time, and finally with the design of a control systems for the application of drones in the last mile delivery of items.

For each problem, after a deep analysis of the related state-of-the-art literature for the setting of the benchmarks to overcome and the choosing of the best resolution method, the related work has been developed also taking into account the company's needs so as to formulate the most complete set of business rules, bridging the gap between companies and academia and satisfying the main requirements of innovations requested by the principles of Industry 4.0. Each problem is thus studied, formalized, and tested taking both the academics' and companies' points of view so as to provide the most complete solution possible. After the examination of the previous issues, the thesis is concluded by presenting a future outlook for logistics in the Industry 4.0 era. It highlights the importance of adaptability and continuous innovation to thrive in this digitally driven landscape.

Overall,it is worth mentioning that this research contributes valuable insights for academics and industry professionals, aiming to navigate the transformation of logistics and supply chain management in the age of Industry 4.0 and has been rewarded of several prizes and publications international distinguished conferences, and journal articles.

Impara da ieri, vivi per oggi, spera per il domani.
La cosa più importante è non smettere mai di farsi domande
Albert Einstein

A me stessa, per la mia tenacia e perseveranza.

Contents

Preface

This thesis is submitted in partial fulfillment of the requirements for the degree of *Philosophiae Doctor* in Electrical and Information Engineering at the *Politecnico di Bari*. This work was supported by ISIRES Istituto Italiano Ricerca e Sviluppo – Organismo di Ricerca S.r.l. as one of the parties of the public-private laboratory IoT 4.0 of Politecnico di Bari, and the italian logistic company the E80 group.

The research presented in this dissertation was conducted at the *Decision and Control Laboratory* of Politecnico di Bari under the supervision of Professors **Mariagrazia Dotoli**, **Graziana Cavone**, and Engr. **Antonio Cerviotti** between November 2020 and October 2023.

This work is to the best of my knowledge original, except where acknowledgments and references are made to previous work. Most part of this thesis has been published during these three years in different scientific publications where I am one of the authors. The papers are preceded by an introductory chapter (Chapter 1), that relates them to each other and provides background information and motivation for the work, and a chapter containing the whole literature analyzed during the research period. A version of Chapters 3 and 7 have been presented in International Journals [1], [2], Chapters 5,6 and 8 have been published in the proceedings of International Conferences [4]–[6], while the work presented in Chapter 4 has been published in the proceedings of an International Conference [7] and extended in a submission for a International Journal [3]. A concluding chapter (Chapter 9) summarizes the main outcomes and findings for future developments.

Professors **Mariagrazia Dotoli** and **Graziana Cavone** were involved in the early stages of concept formation, as well as data collection, numerical implementation, analysis of experiments, and composition of the above-mentioned manuscripts.

Data collection, numerical implementation, analysis of experiments, and preparation of the original version of Chapter 7 was done primarily by Engr. **Silvia Proia** while I contributed mostly in the early stages of concept formation and during the manuscript composition.

The real datasets used for the experiment in Chapters 3,6 and 7 were provided by Engr. **Antonio Cerviotti**, from the E80 Group. Chapter 7 was part of an extended international collaboration with Professor **Slim Hammadi**, Dr. Engr **Haifa Zgaya**, Dr. Engr **Sara Ben-Othman**, Engr. **Hadrien Salem**, from the *Universitè de Lille*, who were involved in the early stages of concept formation, as well as data collection, numerical implementation, analysis of experiments, and composition of the above-mentioned manuscript.

The full list of papers written by the author is reported hereafter.

List of Papers Written by the Author

International Journal Articles

[1] Tresca, G., Cavone, G., Carli, R., Cerviotti, A., and Dotoli, M., "Automating bin packing: A layer building matheuristics for cost effective logistics," *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 3, pp. 1599–1613, 2022.

[2] Tresca, G., Hadrien, S., Cavone, G., *et al.*, "A matheuristic approach for delivery planning and dynamic vehicle routing in logistics 4.0," *IEEE Transactions on Automation Science and Engineering*, vol. **(First revision round)**,

[3] Tresca, G., Cavone, G., Carli, R., and Dotoli, M., "A matheuristics for the configuration of automated vertical lift modules warehouses," *IEEE Transactions on Automation Science and Engineering*, vol. **(to be submitted)**,

International Conference Proceedings

[4] Cavone, G., Carli, R., Troccoli, G., Tresca, G., and Dotoli, M., "A milp approach for the multi-drop container loading problem resolution in logistics 4.0," in *2021 29th Mediterranean Conference on Control and Automation (MED)*, 2021, pp. 687–692. DOI: 10.1109/MED51440.2021.9480359.

[5] Tresca, G., Cavone, G., and Dotoli, M., "Logistics 4.0: A matheuristics for the integrated vehicle routing and container loading problem," in *2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2022, pp. 333–338. DOI: 10.1109/SMC53654.2022.9945179.

[6] Proia, S., Cavone, G., Tresca, G., Carli, R., and Dotoli, M., "Automatic control of drones' missions in a hybrid truck-drone delivery system," in *2023 9th International Conference on Control, Decision and Information Technologies (CoDIT)*, 2023, pp. 1477–1482. DOI: 10.1109/CoDIT58514.2023.10284110.

[7] Tresca, G., Cavone, G., Carli, R., and Dotoli, M., "A mathematical model for the optimal configuration of automated storage systems with sliding trays," in *2023 European Control Conference (ECC)*, 2023, pp. 1–6. DOI: 10.23919/ECC57647.2023.10178251.

Chapter 1

Introduction

Industry 4.0 is an actual trend whose aim is the integration of new technologies to improve working conditions, create new business models, and increase productivity and production quality in industrial plants. The term "Industry 4.0" was established in 2011 by Henning Kagermann, Wolf-Dieter Lukas, and Wolfgang Wahlster, during a speech held at the 2011 Hanover Fair, in which they announced the Zukunftsprojekt Industrie 4.0, a project for the industry of the future that included investments in infrastructure, schools, energy systems, research bodies, and companies to modernize the German production system and bring German manufacturing back to the top of the world making it globally competitive [1],[2]. Nowadays, the concept Industry 4.0 is broadly used worldwide to represent a new industrial revolution based on the use of enabling technologies such as robotics and automation, sensors, connectivity, and programming to advance the industrial sector. As suggested by its name, this is the fourth revolution that the Industry has experienced over the years. Each industrial revolution has brought radical changes to industries' paradigms. In particular:

1. the first industrial revolution occurred in 1760. It is characterized by the invention of the steam engine which led to the introduction of the mechanization of production and tools as the mechanical frames that were driven by the power of water and steam on mechanical equipment. This revolution led to an increase in the ability to produce goods and consequently brought to rapid economic growth;
2. the second industrial revolution, in 1870, regards the use of electricity and oil as energy sources, leading to increased mass production and the introduction of new technologies like the telegraph and telephone;
3. the third industrial revolution, started in 1970s, was driven by information technology and electronics, further enhancing automation and production quality through increasingly powerful computers and the introduction of the first industrial robots;
4. the fourth industrial revolution, Industry 4.0, includes a further evolution of industrial automation. This revolution differs from previous ones by leveraging digital technologies and the IoT. The IoT refers to the extension of the internet to the world of objects and physical locations, enabling connection and interaction between smart devices. In Industry 4.0, machines are interconnected and communicate with each other, performing self-diagnosis and preventive maintenance. This improves the efficiency and reliability of production processes and in particular, the production planning and control activities whose aim is to define what, how much, and when to produce, buy, and deliver so that the company can match manufacturing performance with customer demands [3].

As it emerges from the above discussion, one of the main elements of Industry 4.0 is digitization, in fact, through digitization, production processes become smarter and more flexible. More in detail, the digitization of industrial operations and processes consists in real-time data collection from sensors and connected devices, providing greater visibility and control over production processes. This data can be analyzed and used to optimize operations, predict failures, adopt preventive measures, and personalize products based on individual customer needs [4]. As a matter of fact, with the emergence of Industry 4.0, the traditional factory is transformed into a

smart factory which is characterized by digitized and interconnected production, where processes are optimized through the use of advanced technologies such as artificial intelligence, augmented reality, and 3D printing, and robots and humans work closely together, exchanging information and learning from each other leading to increased production efficiency, flexibility, and product quality. Since 2011, the main topics of Industry 4.0 have been extensively studied and analyzed. For this reason, the new trend of research is slowly moving from Industry 4.0 towards the recent concept of Industry 5.0 where the new milestone is to leverage the creativity of human experts in collaboration with efficient, intelligent, and accurate machines, in order to obtain resource-efficient and user-preferred manufacturing solutions supported by new Information Technologies (ITs) such as edge computing, digital twins, collaborative robots, Internet of every things, blockchain, 6G and beyond networks [5].

The concept of Industry 4.0 has extended also to logistics, leading to Logistic 4.0, which consists in the combination of the aforementioned technologies with the processes of planning, implementing, and controlling the efficient and effective flow of goods, materials, and information from the point of origin to the point of consumption. Moreover, Logistics 4.0 aims at optimizing and improving the management of various activities within the supply chain such as production, inventory management, transportation, warehousing, packaging, and distribution. Effective logistics management requires the use of technology, such as Transportation Management Systems (TMS), Warehouse Management Systems (WMS), and supply chain visibility tools, to track and manage inventory, optimize transportation routes, and streamline operations. In order to concretize the general concept of Logistic 4.0, several problems and operations have been analyzed, studied in their vulnerabilities and weaknesses, and then improved with some of the most promising control systems and optimization methods. This evolution plays a critical role in meeting customer demands, optimizing operational efficiency, reducing costs, enhancing customer satisfaction, and ultimately contributing to the success and competitiveness of businesses in various industries. Among the various classifications of logistics' activities, one of the most commonly used taxonomies considers the following two different branches [6]:

- internal logistics, also known as inbound logistics or intralogistics, deals with the organization, management, and control of physical and information flows within a company. These flows include managing raw material stocks, internal material handling, product distribution, and reverse logistics for handling returns. It involves activities such as inventory management, warehousing, material handling, production planning, and transportation of goods between different departments or production stages. The goal of internal logistics is to optimize processes and reduce costs, ensuring an efficient flow of goods within the company;
- external logistics, also known as outbound logistics, pertains to the transportation and distribution of goods from the company's facilities to the end customers or other external stakeholders. This branch includes activities such as order processing, transportation management, and delivery to the final destination. External logistics often includes working with external partners such as carriers, freight forwarders, and distributors to ensure timely and efficient delivery of products to customers. The focus is on meeting customer demands, optimizing transportation routes, and managing the overall supply chain network beyond the company's own operations.

In summary, internal logistics deals with the internal movement and management of goods within the company's facilities, while external logistics deals with the management and control of transportation and distribution of goods to customers or other external stakeholders. Both internal and external logistics are integral parts of the overall supply chain management, and effective coordination between the two is crucial for seamless operations and customer satisfaction.

The objective of this thesis is consistent with the primary objective of logistics, i.e., to follow the lifecycle of goods from their storage inside the warehouse to their delivery to the final customers, aiming at optimizing the whole process so as to save time and money, and ensuring the correctness of the operations. In particular, in this manuscript, after a detailed analysis of the state of the art on Logistics 4.0 and related problems, specific and challenging internal and external logistics problems are discussed, and innovative control and optimization methods are proposed and tested for their solution.

Part 1 of the thesis focuses on internal logistics and in particular on the optimization and automation of storage systems and related activities. The tackled problems are:

- automation of the bin packing;
- optimal design of Vertical Lift Modules (VLM) warehouses.

Both of such problems basically focus on the optimization of the space occupation, respectively in bins and VLM warehouses. The aim here is to solve them in an automated and time efficient way. As for the former problem, this thesis aims at solving the literature's well-known three-dimensional bin packing problem (3D-BPP) with the addition of several logistic requirements, for its practical application. The 3D-BPP regards the efficient packing of a set of items of various sizes into a minimum number of containers, or "bins", while adhering to certain constraints. It represents one key problem in logistics, that is the optimization of the arrangement of goods within a loading unit, such as a pallet or a container. This thesis proposes a novel matheuristic technique that provides stable and compact bin configurations in less than half a minute per bin on average, facing the high computational complexity of the problem. In fact, the 3D-BPP is known to be NP-hard, which means that finding an optimal solution becomes increasingly difficult as the number of items increases. The proposed approach allows considering compatibility constraints for the items (e.g., final customer and category of the items), and the use of robotized layer picking in automated warehouses. In effect, layers composed by only one type of items (i.e., monoitem layers) can be directly picked and placed on the pallet by a robotic arm without the intervention of any operator. Consequently, the adoption of this approach in warehouses could drastically improve the efficiency of the packing process. As for the second problem, the contribution of this thesis consists in the definition of an innovative matheuristics to support the automated and efficient design of a particular type of automated warehouses named Vertical Lift Module warehouses. A VLM is a closed structure composed of a set of shelvings that house a variable number of sliding trays and a lift-mounted module that handles the trays. Each tray is partitioned into sectors where the items are stored. The items are placed on/retrieved from the trays by the logistic operators through an access bay predisposed in the VLM. Automated warehouses like these are designed to meet any specific need, regardless of industry or commodity category, making them ideal for improving the internal logistics for production or distribution. On the one hand, they follow the principle of *"the goods to the man"*, i.e., the order preparation strategy in which goods arrive directly to the operator through automated systems so as that the stored goods are available and easy-to-pick to the operators and thus leading to a drastic reduction of the effort and the time required for picking an order and to an improvement of the working conditions by reducing injuries caused by heavy loading or unloading of the orders and leading to greater ergonomics and safety in the workplace. On the other hand, VLM warehouses allow for optimizing the total available space by making full use of the available height and thus exploiting the generally unused space. Designing a VLM for logistics companies is a complex and time-consuming task since the current manual approach relies on experienced operators and iterative processes and lacks of specialized software and tools. The main challenge in this operation lies in optimizing space utilization while adhering to various design constraints. In this

context, this thesis proposes a two-phases matheuristic algorithm able to efficiently optimize the whole configuration of the VLM, from the placement of the items to the allocation of the trays into columns. The considered objective is to automate and optimize the VLM design process, taking into account space utilization, logistical constraints, and operational requirements such as weight limits and item rotation, and determine the number of trays and column configurations.

Part 2 focuses on external logistics and in particular on the delivery planning and online replanning problems. More in detail, the tackled problems are:

- the multi-drop Container Loading Problem (CLP);
- the integrated Vehicle Routing Problem (VRP) and Container Loading Problem;
- the delivery planning and Dynamic Vehicle Routing Problem (DVRP);
- the control of a hybrid truck-drone delivery system for the last mile delivery.

As for the first problem, i.e., the multi-drop container loading problem, its purpose in the logistic process is the packing of multiple bins, associated to multiple deliveries to one or more customers, into a finite number of Transport Units (TUs). Differently from the traditional CLP, the multi-drop CLP has been rarely handled in the literature, while effective algorithms to automatically solve this problem are needed to improve the efficiency and sustainability of internal logistics. To this aim, this thesis proposes a novel algorithm that solves a delivery-based mixed integer linear programming formulation of the problem. The algorithm efficiently determines the optimal composition of TUs by minimizing the unused space, while fulfilling a set of geometric and safety constraints, and complying with the delivery allocation.

As for the second tackled problem, i.e., the combined optimization of CLP and of the vehicles' routes, the aim is to support logistic companies in reducing planning times and freight delivery costs, in what is called *"the integrated vehicle routing and container loading problem"* or *"delivery planning"*. As a matter of fact, in delivery planning, given a set of delivery requests, both the routes and load configurations of TUs are to be established. In the literature, this combined problem is defined as Three-dimensional Loading Capacitated Vehicle Routing Problem with Time Windows (3L-CVRPTW). However, these problems are generally tackled separately and referred to as the vehicle routing problem and the container loading problem, respectively. Moreover, only a few contributions present solution approaches for real logistic systems, and these methods are mainly based on heuristics. In this work, a novel matheuristic algorithm is defined for the integrated solution of the vehicle routing problem and container loading problem. Using the predefined model of the container loading problem, the approach aims to minimize the total travel costs and the clients' time windows violations in the routes' definition, while optimizing the configuration of the cargo inside each TU.

As for the third tackled problem, i.e., optimization of the delivery process combined with the implementation of the dynamic vehicle routing problem, the aim is to facilitate the delivery planning and online rerouting in case of unexpected events that may occur during the shipping operations (i.e., traffic congestion, delays at some customers, etc). An algorithm is presented that automatically generates feasible routing and loading plans for a set of TUs, and then updates in real-time the nominal route in case of unexpected events. More specifically, the algorithm is composed of two phases that cooperate together sequentially: the first phase considers the delivery plan described in the second section and thus provides to the second phase the number and type of transport units to be used, the composition of the bins in each transport unit, and the corresponding route that, in case of unexpected events (e.g., accidents, slowdowns, etc.) affecting one or more routes, re-routes the involved trucks guaranteeing the maximum efficiency in regards to travel cost, travel time, and quality of service.

Finally, the last considered problem regards the last-mile delivery (i.e., the problem of transporting from a distribution center or local hub to the end destination, typically a customer's home or a retail store) in a brand new environment related to the smart cities. In particular, a novel problem is addressed that regards the optimal control of the drones' missions in a hybrid architecture that combines the use of a drone and a truck to perform a sequence of pick-ups and deliveries. The drone can perform three different pick-up and delivery missions: truck to point (i.e., pick-up from the truck and delivery to the customer), point to point (i.e., delivery to a customer and pick-up from the subsequent customer), and point to truck (i.e., reentry from a customer to the truck). To accomplish the desired mission, the drone is optimally guided by a receding horizon Linear Quadratic Regulator (LQR) in all its operating modes (i.e., ascent from customer and from truck, and descent to truck mode, free flight with/without payload mode, descent for pick-up/delivery mode). The choice of this particular control strategy is the optimality of the RH-LQR, as a matter of facts it seeks to find a control policy that optimizes a performance criterion over a finite time horizon while taking into account the system's dynamics and constraints. This results in a locally optimal control strategy for the given horizon. This control strategy has several advantages: it can adapt to changes in the system dynamics or constraints, it ensures that the system operates within specified limits, making it suitable for applications where constraints are critical, such as robotics and process control, its designs ensure closed-loop stability for linear time-invariant systems (this means that the controlled system will remain stable under the chosen control law), moreover, it can be applied to a wide range of systems, both continuous and discrete, linear and nonlinear, as long as a suitable model of the system dynamics is available.

All the problems tackled in this thesis are relevant in the industry and their effective solution can lead to several improvements such as savings in time and money, ensuring the optimal of processes, reducing human errors, and, in some cases, correcting them in real-time. The research approach adopted when addressing the various problems started with the study and analysis of the existing literature methods, from the most consolidated to the most recent ones. Based on the literature gaps and the industrial needs novel approaches were implemented and validated. The main challenges faced during those activities were related to the definition and formalization of the wide set of logistic constraints required by the company that were still not completely formalized in the literature (e.g., very few contributions took into account a complete combination of the container loading problem with the vehicle routing problem with time windows, or also the compatibility constraints of the items in the composition of bins), and in the implementation of methods who were able to ensure high-quality solutions in low computation time. For the latest requirement, the focus has been on matheuristics which allows the combination of the advantages of exact solutions and heuristics [7]). Moreover, the work conducted in this PhD thesis has gained interest in the scientific community. In particular, the preliminary version of the layer building BPP solution approach has been awarded the prize "IEEE Italy Section - ABB Master Thesis Award". The subsequent advancements of the research and the corresponding outcomes led to the publication/preparation of three international journal papers [8]–[10] and four conference papers [11]–[14].

The remainder of this thesis is structured as follows: Chapter 2 contains the problem statement and the nomenclature of the problems tackled in this work and a detailed analysis of the most important literature contributions related to the field of interest of this thesis (i.e., bin packing problem, container loading problem, vehicle routing problem, etc.). Then the remaining chapters are dedicated to the various tackled problems and the proposed solutions. The corresponding structure is as follows: the abstract and the introduction give an overview of the topic of the

chapter, then the subsequent sections are dedicated to the mathematical formulations of the problems and to the description of the proposed algorithm, which is then validated in the experimental results section. Finally, the conclusions close the chapter, providing insights on the achieved results and possible extensions of the proposed methodology. More in detail, Part 1 regards internal logistics and is composed by two chapters: Chapter 3 on the automated layer building bin packing and Chapter 4 on the optimization of the Vertical Lift Modules (VLM) warehouses. Differently, Part 2 regards external logistics and includes four chapters: Chapter 5 addresses the multi-drop container loading problem, Chapter 6 combines the container loading problem with the vehicle routing problem, providing a static method for the delivery planning, Chapter 7 joins the delivery planning of the previous chapter with the online replanning of the fleets, while Chapter 8 considers the automatic control of drones and trucks in the last-mile delivery problem. Finally, Chapter 9 provides the conclusions and possible further developments of this doctorate thesis.

References

[1] Ramaa, A., Subramanya, K., and Rangaswamy, T., "Impact of warehouse management system in a supply chain," *International Journal of Computer Applications*, vol. 54, no. 1, 2012.

[2] Lasi, H., Fettke, P., Kemper, H.-G., Feld, T., and Hoffmann, M., "Industry 4.0," *Business & information systems engineering*, vol. 6, no. 4, pp. 239–242, 2014.

[3] Bueno, A., Godinho Filho, M., and Frank, A. G., "Smart production planning and control in the industry 4.0 context: A systematic literature review," *Computers & Industrial Engineering*, vol. 149, p. 106 774, 2020.

[4] Barreto, L., Amaral, A., and Pereira, T., "Industry 4.0 implications in logistics: An overview," *Procedia Manufacturing*, vol. 13, pp. 1245–1252, 2017.

[5] Maddikunta, P. K. R., Pham, Q.-V., B, P., *et al.*, "Industry 5.0: A survey on enabling technologies and potential applications," *Journal of Industrial Information Integration*, vol. 26, p. 100 257, 2022. DOI: https://doi.org/10.1016/j.jii.2021.100257. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2452414X21000558.

[6] Winkelhaus, S. and Grosse, E. H., "Logistics 4.0: A systematic review towards a new logistics system," *International Journal of Production Research*, vol. 58, no. 1, pp. 18–43, 2020.

[7] Fischetti, M. and Fischetti, M., "Matheuristics," in *Handbook of heuristics*, Springer, 2018, pp. 121–153.

[8] Tresca, G., Cavone, G., Carli, R., Cerviotti, A., and Dotoli, M., "Automating bin packing: A layer building matheuristics for cost effective logistics," *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 3, pp. 1599–1613, 2022.

[9] Tresca, G., Cavone, G., Carli, R., and Dotoli, M., "A matheuristics for the configuration of automated vertical lift modules warehouses," *IEEE Transactions on Automation Science and Engineering*, vol. **(to be submitted)**,

[10] Tresca, G., Hadrien, S., Cavone, G., *et al.*, "A matheuristic approach for delivery planning and dynamic vehicle routing in logistics 4.0," *IEEE Transactions on Automation Science and Engineering*, vol. **(First revision round)**,

[11] Cavone, G., Carli, R., Troccoli, G., Tresca, G., and Dotoli, M., "A milp approach for the multi-drop container loading problem resolution in logistics 4.0," in *2021 29th Mediterranean Conference on Control and Automation (MED)*, 2021, pp. 687–692. DOI: 10.1109/MED51440.2021.9480359.

[12] Tresca, G., Cavone, G., and Dotoli, M., "Logistics 4.0: A matheuristics for the integrated vehicle routing and container loading problem," in *2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2022, pp. 333–338. DOI: 10.1109/SMC53654.2022.9945179.

[13] Tresca, G., Cavone, G., Carli, R., and Dotoli, M., "A mathematical model for the optimal configuration of automated storage systems with sliding trays," in *2023 European Control Conference (ECC)*, 2023, pp. 1–6. DOI: 10.23919/ECC57647.2023.10178251.

[14] Proia, S., Cavone, G., Carli, R., and Dotoli, M., "Optimal control of drones for a train-drone railway diagnostic system," in *2023 IEEE 19th International Conference on Automation Science and Engineering (CASE)*, IEEE, 2023, pp. 1–6.

Chapter 2

State of the Art

This Chapter, provides a wide discussion of the literature related to the problems analyzed in this thesis. The examination of the existing milestones of knowledge serves as a foundation for comprehending the landscape in which the research presented in this thesis operates. By surveying and synthesizing the relevant literature, it is possible to gain a comprehensive perspective on the concepts, methodologies, and innovations that have shaped the fields of control and optimization methods applied to Industry 4.0 and Logistics 4.0.

More in detail, the first purpose of this chapter is to provide a context for understanding the lineage and evolution of the problems addressed in this research, tracing the development of ideas and approaches, identifying the key milestones and seminal works that have contributed developing the current state of the art. Secondly, the second aim is to identify gaps, challenges, and unexplored territories within the existing knowledge, pointing the attention toward areas where further investigation and innovation are warranted. Finally, this chapter aims at supporting the validation of the relevance and significance of this research by establishing connections between this research work and the broader academic and industrial landscape, and by evaluating the existing methodologies and solutions. In this way it is possible to better position this research in its broader related context and highlight the novelty and contributions it provides.

Consequently, this chapter contains, in the following sections, a comprehensive review of the state-of-the-art on: (1) the three-dimensional (3D) Bin Packing Problem, (2) the Vertical Lift Modules warehouse optimization problem, (3) the multi-drop multi-container loading problem, (4) the integrated vehicle routing and container loading problem, and (5) the last mile delivery problem.

2.1 The Bin Packing Problem

The bin packing problem definition relies on the Cutting-Stock problem, firstly proposed in [1] and in [2]. In particular, the authors discussed the problem of cutting standard pieces of stock material into pieces of specified sizes while minimizing material waste and the related costs. This problem was then extended to various applications and redefined leading to the broad category of Cutting and Packing (C&P) problems. One of the most famous categories in C&P problems is the class of one-dimensional Single Bin-Size BPP (1D-SBSBPP) and consists in packing a set of strongly heterogeneous items of given weights to a minimal number of bins of identical capacity (i.e., only one bin type) such that for each bin the total capacity of the small items does not exceed the bins' capacity. Further classes identified by the widely accepted typology of [3] are for example the two- and the three-dimensional Single Bin-Size Bin Packing Problem (2D- and 3D-SBSBPP), the Multiple Bin-Size BPP (MBSBPP) if the bins are weakly heterogeneous, and the Residual BPP (RBPP) if the assortment of bins is strongly heterogeneous.

In this thesis, the focus is on the 3D-SBSBPP for two different internal logistic applications, i.e., the automated palletizing of freights presented in Chapter 3 and the definition of the VLMs' configuration presented in Chapter 4), for which several formulations are presented in the literature. The classic ones consider only geometric conditions, i.e., items must not overlap and must lie inside the bins; while more recent formulations include additional conditions of practical utility, such as the stability of the configuration and the family or category of items (i.e., items

of the same family/category have common fundamental characteristics and can be grouped in the same bin; for example, chemical products should not be combined with food). In Table 2.1, similarly to the early classification presented in [4], it is specified for each article discussed in this section the logistic requirements defined in the proposed contribution (i.e., rotation of the items, stability, load bearing, weight, family/category, shape, overhang, robotized layer picking). As it emerges from the following review of the state of the art, the problem is typically formulated by employing linear programming (LP), integer linear programming (ILP), and mixed-integer linear programming (MILP) (see, e.g., the reviews [5], [6], [7]). Nevertheless, the contributions on real logistic applications of the 3D-SBSBPP mainly consider the MILP formulation since it properly allows to represent geometric and safety constraints, see e.g., [8], [9], [10]. As for the resolution of the problem, due to its NP-hard nature, the majority of contributions consider heuristic or matheuristic approaches that allow to obtain good quality solutions.

Table 2.1: Logistic requirements considered in publications on packing problems

Publication	Rotation	Stability	Load Bearing	Weight	Family/ Category	Shape	Overhang	Robotized layer picking	Space sectorialization
Moura and Oliveira (2005) [11]	X	X							
Saraiva *et al.* (2015) [8]			X	X					
Trivella and Pisinger (2016) [9]			X						
Paquay *et al.* (2018) [12]	X	X	X	X		X			
Elhedhli *et al.* (2019) [13]		X	X	X	X	X			
Gzara *et al.* (2020) [14]		X	X	X	X	X			
Zaho *et al.* (2020) [15]		X	X						
Jiang *et al.* (2021) [16]	X								
The automated bin packing [17]	X	X	X	X	X		X	X	
The VLMs configuration optimization [18]	X			X	X				X

One of the first attempt to define and solve the 3D-SBSBPP as an optimization problem was proposed by [19], [5], and [20], all aiming at minimizing the number of filled containers given a set of items. Since these preliminary approaches are impractical for companies, due to long computational times and the presented geometric constraints, several studies have proposed extended formulations and advanced heuristics or adaptive approaches to produce sub-optimal feasible solutions in a reasonable computational time.

2.1.1 The Automated Bin Packing Problem

The formulation of the automated packing problem tailored to the internal logistics requirements can be defined by taking inspiration from several literature contributions, since basically it requires the definition of the cutting-and-packing classical version with the addition of specific logistic constraints. The resulting problem is evidently complex and the related solution is non-trivial. Consequently, the main aims of the literature contributions are on the one hand the definition of a wide set of detailed logistic constraints and on the other hand the definition of efficient methods to minimize the computational time for their solution.

Some examples are provided by the authors of [11] who propose a greedy randomized adaptive search procedure, based on the *wall building* algorithm (firstly proposed in [21]) without formulating the 3D-SBSBPP as a programming problem, but only providing a list of instructions to be executed by operators. The algorithm allows to decompose the 3D-SBSBPP into two sub-problems. The first aims at composing the items into vertical walls, while the second aims at composing the obtained walls into bins; both minimize the free space in the bin. The authors focus on the resolution of the problem for only one container type and multiple heterogeneous items, and consider geometric, stability, and rotation constraints. Differently from [11], the authors of [22] assume the 3D-SBSBPP to be similar to the parallel-machine scheduling problem. The problem is modeled as a MILP problem including only geometric constraints. The authors consider a heuristic procedure to define a new lower bound based on an LP-relaxation of the

MILP problem, which is improved by including valid inequalities based on some similarities with the parallel-machine scheduling problems. Three further important literature contributions about the 3D-BPP are [23], [24], and [25]. The first two present a two-level tabu search where the first-level aims at reducing the number of bins, while the second one optimizes the packing of bins following a procedure based on the interval graph representation of the packing, which reduces the size of the search space. The last contribution aims at solving large BPPs in two and three dimensions. The authors propose a straightforward heuristics based on the container loading problem method following a wall building approach and on a one-dimensional BPP approach applied then for 2D and 3D-BPP taking into account several logistic constraints such as stability, load bearing, and weight.

Subsequently, [8] considers items as physical boxes and therefore pays particular attention to their physical characteristics, including the weight and material of the considered items. Similarly to [11], the authors do not provide a mathematical formulation of the 3D-BPP, but they present an algorithm for the composition of the bins that aims at minimizing the free space in the bins and the total number of composed bins. The approach first builds horizontal layers of identical items and, then, generates packing schemes by greedily loading layers according to a selection criterion. Further, [9] considers the problem of 3D bin packing including both geometric and load bearing constraints. The goal is to find the minimum number of bins ensuring that the loaded items' average mass center falls as close as possible to an ideal point, for example the center of the bin. The authors propose a MILP formulation of the problem and a multi-level local search heuristics for its resolution. The method leads to good results in terms of bin balancing and computational time on literature examples tests.

In two recent works [10] and [12] the authors succeed in providing a complete formulation of the problem and efficient resolution methods for industrial applications. More in detail, [12] defines a mixed integer programming problem able to find a solution to the three-dimensional Multiple Bin Size Bin Packing Problem with bins of different shapes to fit inside an aircraft. For the resolution of this specific problem, three heuristics are adopted, i.e., Relax-and-Fix, Insert-and-Fix and Fractional Relax-and-Fix. Some tests are performed on test cases specifically designed for this type of problem, showing good results of the heuristics. Nevertheless, this work disregards some important aspects such as the compatibility among the different families/categories of items to be packed and the possibility to create configurations with only one type of items both for layers and bins. Such precautions are fundamental in a real company environment and can further reduce the computational time and simplify the picking process of the items from the warehouse.

Further recent contributions [26], [15] and [16] introduce the adoption of new resolution approaches derived from the Internet technology field. The first one uses a hybrid genetic approach for the resolution of the heterogeneous BPP transportation and distribution to various locations by satisfying practical constraints, such as box rotation, fragility, container stability, weight, overlapping, and shipment placement. The second one uses a constrained deep reinforcement learning method for the resolution of an online 3D-BPP formulated as a constrained Markov decision process which includes physical stability and rotation constraints, while the last one exploits multimodal deep reinforcement learning in order to reduce the computational complexity of the classical version of the BPP (i.e., without the logistic constraints) and solve medium-scale instances of 100 items while most existing methods are only able to handle up to 50 boxes in short computational times.

Further particularly recent approaches are presented in [13] and [14], which address the mixed-case pallettization problem, i.e., an extension of the classic 3D-BPP that incorporates practical logistic features, such as bin stability, item support, family/category groupings, isle friendliness, and load bearing. To solve such a problem, a column-generation solution approach is proposed, where the pricing sub-problem is a two-dimensional layer-generation problem.

2.1.2 The VLMs Configuration Problem

Vertical lift module warehouses are a recent and versatile automated solution whose innovative structure, particularly suitable for items of small dimensions, allows a higher level of productivity with respect to other systems based only on trays (e.g., carousels). In fact, the presence of a picking bay allows for reducing operators' walking, even if the replenishment is still carried out manually [27]. According to Battini *et al.*[28] there are multiple advantages to the use of VLM in automated warehouses such as the higher volume utilization with respect to classical warehouses, the reduction of damages in the storing of products, the reduction of distances traveled by operators, lower labor costs, greater control over inventory, provision of an ergonomic solution, as it results in improved working conditions for operators, while also reducing the potential cost associated with injuries, and ensuring safe storage of products, preventing their possible theft or damage. On the other hand, VLMs also have some weaknesses, such as the higher, potential downtime for the picker, who must wait for the current tray to be stored and the next one to be retrieved. To improve such idle time, modifications are being developed on VLMs, such as the optimization of the positioning and placement of items in columns that can lead to higher system throughput. Some allocation strategies for this kind of warehouse were studied by Roodbergen *et al.* [29] and can be classified as follows:

- *dedicated allocation*: each product type is allocated to a fixed location. The replenishment of that product always takes place at the same location.
- *random allocation*: all locations have the same probability of receiving trays or products, for each type of load;
- *nearest free allocation space*: the first available empty location is used to store products;
- *full-turnover storage*: the locations are determined based on the frequency of load demand. Frequently demanded products are allocated close to the pick-up points, otherwise at more distant locations. The turnover frequencies must necessarily be known in advance;
- *class-based storage assignment*: this method splits the available space in the warehouse into areas. Each load can be assigned to a location through priority criteria, such as frequency of demand. To implement a class-based storage approach, one must consider:

  1. area division of the warehouse, i.e., determining the number of classes;
  2. area dimensioning, i.e., determining the number of products to be allocated in an area;
  3. product allocation, i.e., determining in which area to allocate each type of product.

In general, besides the type of strategy implemented, VLMs are more difficult to reconfigure than traditional storage systems and have higher maintenance costs since the composition of the tray is predetermined [30]. For this reason, the optimal planning of the items' position inside a VLM is a crucial operation in the optimization of the entire warehouse both in terms of time and cost. Moreover, the problem of optimal storage allocation is considered NP-hard due to the large number of model parameters considered such as items' characteristics and their relationships, different layouts of the trays, grouping, placing, and routing strategies. In the related literature, Meller and Klote [27] discuss the advantages of modeling the throughput of VLMs, particularly in supporting the respective design processes, and provide different models for its computation and evaluation in a VLM pod, i.e., the combination of multiple modules. Also Dukic *et al.* [30] focus on the throughput computation, but extend their study to dual tray VLMs. This problem is also discussed in [31], [32] using different optimization models and economic evaluations are provided. Additional works related to VLMs' throughput optimization regard order batching, as discussed in [32], where the aim is to optimize the grouping of items' orders, so as to minimize

the picking time and total completion time in VLMs. Finally, in the study by Rosi *et al.* [33] a simulation based model is developed for the performance analysis of a single-tray VLM with various configurations. As it emerges, the literature on VLMs is particularly limited and it is generally focused on the optimization of VLMs throughput and proposing solutions that aim at minimizing trays' movements [28]. Differently, little attention is devoted to the formulation and solution of the VLM internal configuration design problem.

As anticipated in the previous section, for the problem formulation, a well-know class of models that could be adapted to the optimal allocation of items inside a tray and of trays inside the VLM is represented by the Cutting and Packing (C&P) problems whose general aim is to pack a given set of items with different sizes into the minimum number of containers can be adapted to the optimization of the space within a tray and a VLM. Also in this case, it has to be highlighted that although it might be possible to set up an effective mathematical programming (MP) model and to solve it with general-purpose software, various logistic requirements can be hardly formulated and included in a unique MP model. Then, for the solution of the problem an alternative effective approach, as highlighted by the literature [34], consists in using the MP solver as a basic tool within a heuristic framework. This combination leads to the matheuristic approach, where the heuristic is built around the MP model. Matheuristics has recently gained attention, as proved by the various publications [35], [36] and dedicated sessions in related conferences, due to its effectiveness in efficiently solving complex problems. In fact, various applications of matheuristics are present in the literature such as packing problems, windfarm layout optimization, vehicle routing, etc. [34].

2.2 The Delivery Management Problems

As for external logistics, the research on distribution problems is attracting ever-growing attention, driven by the rising demand for freight transport, particularly from distribution centers to final customers. The main challenges within this domain are mainfolds most challenging are related to the definition of efficient load plans and the dynamic routing of vehicles in real-time and in a variable scenario is even more complex. As a matter of facts, efficient load planning remains a non-trivial activity in logistics. It involves the delicate art of optimizing the arrangement of goods within transportation units, such as containers, trucks, or vans. Achieving this optimization is critical not only for minimizing transportation costs and times but also for reducing the environmental footprint of logistics operations. The literature related to load planning encompasses a range of techniques, from heuristic and metaheuristic approaches to mathematical programming models, each aimed at addressing the complexities of item packing and spatial constraints. On the other hand, dynamic vehicle routing (i.e., the routing which operates particularly in real-time or in a dynamic environment), presents its own set of implementation challenges. This problem domain involves adapting the routes and schedules of delivery vehicles on the fly, and responding to dynamic factors such as traffic conditions, weather, changing customer demands, and unexpected disruptions. The effective resolution of dynamic routing problems is crucial for ensuring timely deliveries, optimizing fuel consumption, and providing exceptional service to customers. In addition to these challenges, a particular application field has been taken into account which considers the integration of drones into the logistics and distribution network. This is motivated by the fact that numerous studies have emphasized the importance of drones and their flexibility in various applications, from agriculture to disaster management. Drones offer unique advantages, such as high speed, low energy consumption, and the ability to navigate in three dimensions, making them invaluable in scenarios where traditional vehicles face limitations. However, in external logistics, drones have their own constraints, such as limited delivery range and payload

capacity. Therefore, the combination of drones and trucks for last-mile deliveries holds the potential to enhance the efficiency of the entire process.

In the next sections of the chapter, it is presented the literature that addresses these pivotal aspects of external logistics and distribution. In particular, various methodologies and algorithms developed to tackle load planning, dynamic routing, and truck-drone systems are explored. This knowledge provides a strong foundation for the subsequent chapter of the thesis, aiming at strengthen the innovation of the research contributions and the solutions proposed.

2.2.1 The Container Loading Problem

As it can be evicted from the analysis conducted until now, the BPP is often referred to CLP whose contribution is used for solving the BPP, in fact, the title of the milestone by Chen [19] is *"An analytical model for the container loading problem"*. In the literature, the classical container loading problem is commonly defined as the problem of finding the optimal packing of a set of bins, known as boxes, into containers, so that bins do not overlap and lie entirely inside the containers, while minimizing the non-occupied space, since the main aim is to pack rectangular objects into bigger rectangular boxes. The resulting optimization problem thus focuses on the spatial arrangement of bins in the containers, considering only geometric constraints [37]. On the other hand to make the proposed solutions relevant to real-world applications and differentiate it from the BPP, several further variants of the CLP have been defined.

According to Bortfeldt and Wäscher [38], CLPs can be classified into two different categories according to the definition of the objective function: "input value minimization problems", in which enough containers are available to accommodate all small items, and "output value maximization problems", in which only a subset of the small items can be packed since the availability of containers is limited. In the first class of problems there are Single Stock-Size Cutting Stock Problem, Multiple Stock-Size Cutting Stock Problem, Residual Cutting Stock Problem, Single Bin-Size Bin Packing Problem, Multiple Bin-Size Bin Packing Problem, Residual Bin Packing Problem and Open Dimension Problem, while the second class of problem is composed by Identical Item Packing Problem, Single Large Object Placement Problem, Multiple Identical Large Object Placement Problem, Multiple Heterogeneous Large Object Placement Problem, Single Knapsack Problem, Multiple Identical Knapsack Problem and Multiple Heterogeneous Knapsack Problem. Both categories have different typologies that vary according to the heterogeneity, number, and dimensions of boxes and containers.

Moreover a broad well-known classification of the literature contributions is based on the number of containers included in the optimization problem, thus distinguishing the single- and multi-CLP. Weight and static stability are commonly considered in the single-container case, while dynamic stability has received less attention to date [4]. In most cases, heuristic [11], [39], [40] and meta-heuristic [41]–[43] approaches have been followed, while mathematical models and exact algorithms have usually addressed only basic problems [44]–[47].

A second main classification of the literature contributions on the CLP can be made by considering the number of delivering destinations associated to the bins, thus distinguishing between mono- and multi-drop CLP [48]. Some heuristic [49], [50] and exact [48] algorithms have been proposed to arrange customer bins partitioned sections of the container. However, in the cited works [48]–[50] only one container is considered; moreover, all the constraints – except the geometric and the Last-In-Fist-Out (LIFO) constraints – are generally neglected.

Specifically, the multi-drop scenario can be addressed in two ways. On the one hand, the CLP is integrated with the vehicle routing problem: this results in the combined optimization of the loading of bins into containers and the routing of containers along paths aimed at serving different destinations with the minimum traveling cost. Such an approach suffers from a high

combinatorial complexity as shown in [51]–[53]. On the other hand, for the sake of reducing the problem complexity, the route of delivering operations is commonly assumed to be known in advance. Thus, in the multi-drop scenario, the resulting optimization problem consists in enhancing the CLP with a LIFO policy aimed at fulfilling the loading and unloading of containers.

Although the CLP is of practical applicability, only recently authors have focused on industrial applications, thus better formalizing logistic constraints ([54], [55], [56]) even though most of the contributions still require for integration of important constraints, such as vertical and horizontal stability constraints, load balancing constraints, and LIFO (Last In-First Out) ordering constraints. An innovative version of CLP is proposed by Castellucci *et al.* [54] who consider the schedule of arrival for the boxes and the departure time of the trucks and design a dynamical programming framework which handles the geometric and time characteristics of the problem separately. Further, Xiang *et al.* [55] present a mathematical loading model whose goal is maximizing the 3D space utilization and develop it by a dedicated placement heuristic integrated with a dynamical space division method. Or also, Layeb et al[56] propose a resolution procedure for the single CLP with additional constraints to deal with realistic situations using a new greedy two-step look-ahead procedure that selects the free space deterministically followed by a block search. Finally, note that in Lai *et al.* [49], who propose a heuristic graph-based approach, and in Junqueira *et al.* [48], who formulate a mixed integer linear programming model for multi-customer delivering, only one container is considered; moreover, all the constraints – except the geometric and the multi-drop constraints – are neglected. Nevertheless, the CLP contributions still require for integration of logistic constraints which are barely considered, such as vertical and horizontal stability constraints, load balancing constraints, and LIFO ordering constraints. Moreover, CLPs are mainly considered as problems detached from the planning of delivery routes.

2.2.2 The Combined Container Loading and Vehicle Routing Problem

The second class of combinatorial optimization problems related to external logistics analyzed in this thesis is the one related to the planning of routes whose aim is to compute a set of transportation routes to serve a set of customers with a limited number of vehicles. This problem is generally known as Vehicle Routing Problem (VRP) whose objective is to minimize the cost of the routes while satisfying transportation constraints [57]. The seminal version of this problem was the so-called truck dispatching problem formulated in the 50s' by Dantzig *et al.* [58], who considered a set of distributed customers with their associated demand, and a fleet of vehicles starting from a central depot. The objective is to find the minimal travel cost (e.g., distance or time), such that each customer is visited and served by a vehicle exactly once. Only later in 70s', Golden *et al.* coined the term "Vehicle Routing"in [59], which is a more general definition of the problem. An exhaustive classification of the VRP typologies was proposed by Bai *et al.* in [60] who classify the VRP formulation according to four main classes of features: scenario, (i.e., number of stops/customers, customer service demand quantity, request times, and onsite service/waiting times), time limits (i.e., time window restrictions and travel time), information (i.e., the evolution of information, quality of information, and availability of information) and data. Moreover, they add a further variant to the classification that is the one related to the Automated Guided Vehicle (AGV) routing, which regards the driver-less transport system used in logistics warehouses and marine container terminals. In particular, based on the type of information that characterizes the route, the problem can be classified into static VRP and dynamic VRP and both can be deterministic or stochastic depending on the uncertainty of available information. The static VRP is obviously more suitable for route planning, while the dynamic VRP is more appropriate for real-time re-routing purposes.

As for delivery planning purposes, the static VRP and CLP can be combined leading to the

definition of the so-called Three-Dimensional Loading Capacitated Vehicle Routing Problem. To the best of available knowledge, there are not so many contributions that present a complete formulation of the 3L-CVRP with time windows. The works by Moura and Olivera [61] and [62] present two detailed mathematical formulations: one for the VRPTW and one for the CLP, and solve the problem using a genetic algorithm (GA). The framework integrates these two problems using two different resolution approaches. The first one treats the problem in a sequential approach, while the second uses a hierarchical approach. A more recent work about the 3L-CVRP has been presented by Rojas *et al.* [63], who explicitly explore the impact of three-axes rotation of the bins on load capacity optimization. They develop a two-phase approach: the first phase converts the 3L-CVRP demand into Capacitated Vehicle Routing Problem (CVRP) demand with a heuristic and then finds the solution for the CVRP problem; the second phase aims to obtain the loading of the vehicle with heuristics using rules to obtain the rotation of the items. Moreover, Rajaei et al. [64] not only propose a complete column generation-based heuristic algorithm for the resolution of the 3L-VRPTW but present a detailed analysis on the literature' most used features such as heterogeneous vehicles, the possibility to split deliveries, time window constraints (this thesis consider the opening time as a soft constraint and the closing time as a hard constraint), geometric bin feasibility, bin's and transport units' weight limit, weight distribution on a transport unit, bins' orientation, bin' stacking and stability, reachability, allocation-connectivity and allocation-separation (e.g., the possibility to split or not the cargo according to the customer considered). Other recent works are from Krebs [65], Küçük [66], Meliani[67], and Wang [68], that proposes different heuristic approaches such as the adaptive large neighborhood search, clustering-based techniques or genetic algorithm.

2.2.3 The Dynamic Vehicle Routing Problem

As for the real-time re-routing of deliveries, the dynamic VRP (DVRP) can be more appropriate than the static approach. Firstly introduced in the 80s' by Psaraftis *et al.* [69], the DVRP reveals of particular interest to improve the efficiency of transportation service. In fact, according to the recent surveys by Psaraftis *et al.* [70], and by Pillac *et al.* [71], this kind of problem enriches the classical version of the VRP considering the evolution of route information during its execution and requires to dynamically update an offline scheduled route depending on the real-time state of the transportation. As highlighted by Psaraftis *et al.* [70] in the related literature the dynamic nature of VRP mainly regards changes in customer locations and or demands, while only a limited number of contributions consider changes in travel and/or service times and changes in vehicles availability. In fact, some recent contributions discussing the second and third class of dynamic variations are [72], [73], and [74] consider the online re-routing problem in a dynamic context. In particular, [72] considers dynamic requests and dynamic travel times and implements a communication strategy between drivers and a central dispatch office in order to perform the re-routing of the vehicles, [73] consider the possibility of having service disruptions due to vehicle breakdowns while performing the real-time VRP with time windows and pickup/delivery service, and manage it by considering the minimization of service cancellation and disruption cost in their dynamic programming based algorithm, while [74] implements a knowledge-based approach named PAM (disruption-handling Policies, local search Algorithm, and object-oriented Modeling) for the handling of unexpected events during the urban distribution process. Moreover, in the field of VRP several works use some machine learning and artificial intelligence algorithms to solve the problem and use it in real-world simulations. For example, Xiang *et al.* [75] introduce a demand coverage diversity based metaheuristic in the framework of an ant colony algorithm and use a demand coverage diversity adaptation method in order to maintain the diversity of covered customers in routes. Okulewicz and Mandziuk,[76] used the DVRP as a test problem to investigate

the hypothesis that in case of hard optimization problems, the selection of the proper search space is more relevant than the choice of the solving algorithm. In order to do that they implemented a continuous optimization approach that can be parametrized with different optimization methods such as Particle Swarm Optimization (PSO) and Differential Evolution (DE), and then compare the solution achieved with the ones obtained with the Genetic Algorithm (GA). Frohner *et al.* propose in [77] a white box linear regression model and a neural network-based black box model whose aim is to predict the average time needed to deliver an order for a given time and day, using historic route data collected over three months and Ng *et al.* [78] focus their attention on the problem of inefficient vehicle routing caused by traffic congestion and implement multiple colonies artificial bee colony algorithm for a capacitated VRP able to perform re-routing strategies in case of traffic conditions so that the risk of late delivery is reduced. Sabar *et al.* [79] suggest the use of a self-adaptive evolutionary algorithm because the solution scattered over the search space computed in a parallel manner can better identify the dynamic changes. Due to the complexity of the CLP, VRP, and DVRP, and given that fast computation time is fundamental in their applications, especially for the real-time management of delivery routes, the majority of solution approaches are heuristics such as tabu search, genetic algorithms, ant colony optimization, particle swarm optimization, Markov decision processes, dynamic programming-based approaches, etc. [70]. In particular, the most considered methods are: tabu search, neighbourhood search, insertion methods, nearest neighbour, column generation, genetic algorithms, ant colony optimization, particle swarm optimization, waiting-relocation strategies, Markov decision processes, dynamic programming-based approaches, neural networks, and queuing-polling strategies. It has to be highlighted that the effectiveness of the used heuristics largely depends on the specific considered instance. As for dynamic VRP, according to [70], [80], in the case of variable customers requests the most appropriate heuristics are: waiting-relocation strategies, Markov decision processes, and dynamic programming-based approaches. Differently, in the case of changes in travel and/or service time, according to [80] both ad-hoc heuristics, based on reoptimization, and evolutionary algorithms, in particular genetic algorithms, are mainly appreciated.As a matter of fact, several reviews highlight that evolutionary algorithms are largely employed in solving online problems characterised by stochasticity in the related events, since generally they are fast and provide simple rules to generate solutions of good quality. In addition, also several artificial intelligence methods are used in this field, but according to Zhang [81] the potentiality of the GA led it to be the most promising method since it can approach the optimal solution and shorten the operation time, thus taking both the operation time and efficiency into account.

2.2.4 Drones' Based Last Mile Delivery Problem

Numerous studies highlight the importance of drones and their flexibility in different applications such as agriculture, logistics, disaster management, infrastructure, and many others. Thanks to their high speed, low energy consumption, lightweight, and ability to move in three dimensions, instead of along a discrete set of static roadways, drones can employ paths that are closer to straight-line connections and can circumvent traffic congestion or accidents [82]. However, in the external logistic sector, drones present some limits if compared to trucks, e.g., short delivery range, low supported weight, and limited capacity. Consequently, the combined use of drones and trucks for last-mile deliveries can bring several improvements with respect to the separate use of drones and trucks [83]. In the related literature, the majority of contributions deal with the strategic design of hybrid truck-drone delivery architectures and the offline planning of both trucks and drones tasks. In fact, the recent surveys by Chung *et al.* [83] and by Madani *et al.* [84] highlight that the existing works mainly present mathematical models aimed at offline planning. and they can be roughly divided into traveling salesman problems with drones (TSPD) and vehicle

routing problems with drones (VRPD). For instance, Weng *et al.* [85] focus on deliveries in smart cities where parcels have to be delivered in restricted traffic zones aiming at determining the path of the truck out of the restricted zone and the path of the drone inside the restricted zone minimizing the execution time of the delivery. Similarly, Wang *et al.* [86] propose a novel routing and scheduling algorithm, referred to as hybrid truck-drone delivery, to simultaneously employ trucks, truck-carried drones, and independent drones to construct a more efficient truck-drone parcel delivery system. As it can be deduced, articles that focus on the online control of drone missions in a hybrid truck-drone delivery system are lacking, although this aspect is crucial for the successful completion of parcel delivery. In particular, the efficient control of the whole mission of the drone and the landing on a moving platform (i.e., the truck) are non-negligible problems. The majority of contributions focus on landing on a static platform, while a limited number of papers tackle the problem of landing on a dynamic platform. Promising contributions in this regard are presented in Paris *et al.* [87] and in Falanga *et al.* [88] where the control strategy largely relies on the use of artificial vision, while not specifically focusing on hybrid truck-drone delivery systems applications. In general, the optimal control of drones involves various techniques and algorithms, depending on the specific application and requirements. Some of the most commonly used control techniques for drones include: Proportional-Integral-Derivative (PID) control that is a widely used method for stabilizing drones. It adjusts the drone's orientation by considering the error between the desired and actual orientation angles [89],[90]; Model Predictive Control (MPC) that uses a dynamic model of the drone to predict its behavior and computes control inputs to optimize a future trajectory. MPC can handle complex constraints and is often used in autonomous navigation [91], [92], nonlinear control techniques, are used to handle the complex dynamics of drones, especially in challenging environments; sliding mode control techniques that provide robust control by creating a "sliding surface" where the error converges to zero. It is effective for handling disturbances and uncertainties; or also Linear-Quadratic Regulator (LQR), i.e., an optimal control technique that minimizes a cost function to determine control inputs. It's often used for trajectory tracking and stabilization [93],[94],[95]. The choice of the control technique depends on the specific application, the drone's dynamics, environmental factors, and the level of autonomy required. Many modern drone systems use a combination of these techniques to achieve the desired performance and functionality. This thesis presents a novel control technique for the last-mile delivery problem, where a drone and a truck are able to autonomously collaborate in order to improve the efficiency of the delivery. The proposed approach uses the receding horizon LQR to control the drone in the dynamic landing, managing the real-time changes of the positions of the landing point. The choice of the LQR is motivated by the fact that LQR Control is chosen for its ability to provide optimal control inputs, ensure system stability, and handle disturbances and uncertainties. It is particularly effective for linear systems, relatively easy to implement, and allows for fine-tuning control performance to meet specific requirements. LQR Control is commonly used for regulation and stabilization tasks and can be applied to systems by linearizing around an operating point. However, it may not be suitable for highly nonlinear systems. On the other hand LQR with a sliding horizon combines the benefits of optimal performance, adaptability, constraint handling, predictive capability, and robustness in real-time control applications. It is particularly effective for controlling systems with complex dynamics and nonlinearities, making it a valuable choice for trajectory tracking and dynamic applications that require continuous adaptation and predictive control.

2.3 Logistics' Basics and Nomenclature

Before detailing the different contributions developed during the Ph.D. and thus the different chapters, this section is dedicated to the description of the main basic concepts and the

nomenclature of the logistic contest considered in all the analyzed problems. In general, this thesis aims at automatizing the different processes that are performed with goods, starting from their storing in the warehouses to their delivery to the final customers. Once retrieved from the warehouse, the goods are packed inside the minimum number of identical shipping bins, which are then placed in the proper order inside the transportation means, both in terms of space and arrival at the final customer.

It is assumed that items are goods to be delivered and packed in basic rectangular unit loads, i.e., packages. Differently, bins or shipping bins are a set of items packed in second level unit loads, i.e., pallet loads. The bin can be composed manually by expert operators, or automatically by anthropomorphic robots that can handle single items or homogeneous sets of items organized in layers (i.e., the so-called robotized layer picking). With the aim of facilitating the bin assembly procedure by both operators and robots and reducing economic losses for the company, here it is assumed that a bin is composed of stacked layers, which can be either homogeneous (i.e., containing only one type of items) or heterogeneous (i.e., containing different types of items). Homogeneous layers are further distinguished into monoitem layers, i.e., composed by items with the same identification number (ID), and monocategory layers, i.e., composed by the same category of goods. Conversely, heterogeneous layers are defined as mixed layers that can include items with different ID and category (e.g., food, chemical products, etc.), given that the categories are compatible. A stability index is associated with each item, and such index ranges from 1 to 100: the higher the value, the more stable the item (note that the value is computed based on the geometrical features of the item). In order to have a stable and compact configuration of the pallet it is imposed that items are stacked by a decreasing value of the stability index starting from the bottom layer up to the top one.

The dimensions of the base of bins depend on the pallet dimensions plus a tolerance excess band both in width and length called overhang, while the bins' maximum height depends on the height of the unit load used to transport the bins, namely the Transport Unit (TU). A bin generally contains multiple items, here assumed related to a single delivery, while one or multiple deliveries compose a shipment order. In the loading of the bins inside the TUs it is considered the overhang as the new length (or width) of the bin. In order to load them without compromising their integrity during the transport, the aim is to balance the cargo both when the TU is moving (so bins are not allowed to fall) and when they are still (so upper bins must not smash the lower ones) considering the whole weight of the configuration (e.g., items and pallet) and also the bins are associated with a corresponding stability index. It is assumed that the items are associated to a single shipment order but they may actually be associated to different delivery orders. Actually, depending on the type of shipment, this can include one or more customer deliveries, while each delivery is associated only to a single customer. Consequently, the items have to be grouped by delivery, ID, and category.

As for the shipment, it is supposed that each company has an infinite fleet but a finite number of TUs available. Each TU differs from the other by its dimensions (i.e., length, width and height), and by the maximum weight that it can support. It is preferable that the TUs arrive at each client within a predetermined time window and contain the whole cargo corresponding to each client, even though some exceptions can be eventually managed.

The nomenclature of the problems considered in this thesis and the corresponding meanings are summarized in Table 2.2.

References

[1] Kantorovich, L. V., "Mathematical methods of organizing and planning production," *Management science*, vol. 6, no. 4, pp. 366–422, 1960.

Table 2.2: Logistics' problems nomenclature

Name	Description
Item	object representing goods packed in a basic unit load
Layer	set of items assembled on the same plane
Monoitem layer	layer composed by items with the same ID
Monocategory layer	layer composed by items with the same category
Mixed layer	layer composed by items with different IDs and categories
Bin	set of stacked layers packed in a pallet load
Pallet	wooden platform where items are stacked to form a bin
Delivery	set of items to be transported to one single customer
Shipment	the entire set of deliveries
Category	class of items sharing specific characteristics (e.g., liquids, food, chemical products)

[2] Gilmore, P. C. and Gomory, R. E., "A linear programming approach to the cutting-stock problem," *Operations research*, vol. 9, no. 6, pp. 849–859, 1961.

[3] Wäscher, G., Haußner, H., and Schumann, H., "An improved typology of cutting and packing problems," *European journal of operational research*, vol. 183, no. 3, pp. 1109–1130, 2007.

[4] Bortfeldt, A. and Wäscher, G., "Constraints in container loading–a state-of-the-art review," *European Journal of Operational Research*, vol. 229, no. 1, pp. 1–20, 2013.

[5] Martello, S., Pisinger, D., and Vigo, D., "The three-dimensional bin packing problem," *Operations research*, vol. 48, no. 2, pp. 256–267, 2000.

[6] Valério de Carvalho, J. M., "Using extra dual cuts to accelerate column generation," *INFORMS Journal on Computing*, vol. 17, no. 2, pp. 175–182, 2005.

[7] Brandao, F. and Pedroso, J. P., "Bin packing and related problems: General arc-flow formulation with graph compression," *Computers & Operations Research*, vol. 69, pp. 56–67, 2016.

[8] Saraiva, R. D., Nepomuceno, N., and Pinheiro, P. R., "A layer-building algorithm for the three-dimensional multiple bin packing problem: A case study in an automotive company," *IFAC-PapersOnLine*, vol. 48, no. 3, pp. 490–495, 2015.

[9] Trivella, A. and Pisinger, D., "The load-balanced multi-dimensional bin-packing problem," *Computers & Operations Research*, vol. 74, pp. 152–164, 2016.

[10] Paquay, C., Schyns, M., and Limbourg, S., "A mixed integer programming formulation for the three-dimensional bin packing problem deriving from an air cargo application," *International Transactions in Operational Research*, vol. 23, no. 1-2, pp. 187–213, 2016.

[11] Moura, A. and Oliveira, J. F., "A grasp approach to the container-loading problem," *IEEE Intelligent Systems*, vol. 20, no. 4, pp. 50–57, 2005.

[12] Paquay, C., Limbourg, S., Schyns, M., and Oliveira, J. F., "Mip-based constructive heuristics for the three-dimensional bin packing problem with transportation constraints," *International Journal of Production Research*, vol. 56, no. 4, pp. 1581–1592, 2018.

[13] Elhedhli, S., Gzara, F., and Yildiz, B., "Three-dimensional bin packing and mixed-case palletization," *Informs Journal on Optimization*, vol. 1, no. 4, pp. 323–352, 2019.

[14] Gzara, F., Elhedhli, S., and Yildiz, B. C., "The pallet loading problem: Three-dimensional bin packing with practical constraints," *European Journal of Operational Research*, vol. 287, no. 3, pp. 1062–1074, 2020.

[15] Zhao, H., She, Q., Zhu, C., Yang, Y., and Xu, K., "Online 3d bin packing with constrained deep reinforcement learning," *arXiv preprint arXiv:2006.14978*, 2020.

[16] Jiang, Y., Cao, Z., and Zhang, J., "Solving 3d bin packing problem via multimodal deep reinforcement learning," in *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*, 2021, pp. 1548–1550.

[17] Tresca, G., Cavone, G., Carli, R., Cerviotti, A., and Dotoli, M., "Automating bin packing: A layer building matheuristics for cost effective logistics," *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 3, pp. 1599–1613, 2022.

[18] Tresca, G., Cavone, G., Carli, R., and Dotoli, M., "A mathematical model for the optimal configuration of automated storage systems with sliding trays," in *2023 European Control Conference (ECC)*, 2023, pp. 1–6. DOI: 10.23919/ECC57647.2023.10178251.

[19] Chen, C. S., Lee, S. M., and Shen, Q. S., "An analytical model for the container loading problem," *European Journal of Operational Research*, vol. 80, no. 1, pp. 68–76, 1995.

[20] Faroe, O., Pisinger, D., and Zachariasen, M., "Guided local search for the three-dimensional bin-packing problem," *Informs journal on computing*, vol. 15, no. 3, pp. 267–283, 2003.

[21] George, J. A. and Robinson, D. F., "A heuristic for packing boxes into a container," *Computers & Operations Research*, vol. 7, no. 3, pp. 147–156, 1980.

[22] Hifi, M., Kacem, I., Nègre, S., and Wu, L., "A linear programming approach for the three-dimensional bin-packing problem," *Electronic Notes in Discrete Mathematics*, vol. 36, pp. 993–1000, 2010.

[23] Crainic, T. G., Perboli, G., and Tadei, R., "Ts2pack: A two-level tabu search for the three-dimensional bin packing problem," *European Journal of Operational Research*, vol. 195, no. 3, pp. 744–760, 2009.

[24] Crainic, T. G., Perboli, G., and Tadei, R., "Extreme point-based heuristics for three-dimensional bin packing," *Informs Journal on computing*, vol. 20, no. 3, pp. 368–384, 2008.

[25] Mack, D. and Bortfeldt, A., "A heuristic for solving large bin packing problems in two and three dimensions," *Central European Journal of Operations Research*, vol. 20, no. 2, pp. 337–354, 2012.

[26] Sridhar, R., Chandrasekaran, M., Sriramya, C., and Page, T., "Optimization of heterogeneous bin packing using adaptive genetic algorithm," in *IOP Conference Series: Materials Science and Engineering*, IOP Publishing, vol. 183, 2017, pp. 12–26.

[27] Meller, R. D. and Klote, J. F., "A throughput model for carousel/vlm pods," *IIE transactions*, vol. 36, no. 8, pp. 725–741, 2004.

[28] Battini, D., Calzavara, M., Persona, A., and Sgarbossa, F., "Dual-tray vertical lift modules for fast order picking," 2016.

[29] Roodbergen, K. J. and Vis, I. F., "A survey of literature on automated storage and retrieval systems," *European journal of operational research*, vol. 194, no. 2, pp. 343–362, 2009.

[30] Dukic, G., Opetuk, T., and Lerher, T., "A throughput model for a dual-tray vertical lift module with a human order-picker," *International journal of production economics*, vol. 170, pp. 874–881, 2015.

[31] Calzavara, M., Sgarbossa, F., and Persona, A., "Vertical lift modules for small items order picking: An economic evaluation," *International Journal of Production Economics*, vol. 210, pp. 199–210, 2019.

[32] Lenoble, N., Frein, Y., and Hammami, R., "Optimization of order batching in a picking system with a vertical lift module," in *International Conference on Information Systems, Logistics and Supply Chain*, Springer, 2016, pp. 153–167.

[33] Rosi, B., Grašić, L., Dukić, G., Opetuk, T., and Lerher, T., "Simulation-based performance analysis of automated single-tray vertical lift module," *International journal of simulation modelling*, vol. 15, no. 1, pp. 97–108, 2016.

[34] Fischetti, M. and Fischetti, M., "Matheuristics," in *Handbook of heuristics*, Springer, 2018, pp. 121–153.

[35] Voss, S., Stutzle, T., and Maniezzo, V., *MATHEURISTICS: Hybridizing metaheuristics and mathematical programming*. Springer, 2009.

[36] Fischetti, M. and Lodi, A., "Heuristics in mixed integer programming," *Wiley encyclopedia of operations research and management science*, 2010.

[37] Bischoff, E. E. and Ratcliff, M., "Issues in the development of approaches to container loading," *Omega*, vol. 23, no. 4, pp. 377–390, 1995.

[38] Bortfeldt, A. and Wäscher, G., "Container loading problems: A state-of-the-art review," *Working Paper Series*, 2012.

[39] Scheithauer, G., Terno, J., Riehme, J., and Sommerweiss, U., "A new heuristic approach for solving the multi-pallet packing problem," *Dresden: Technische Universität Dresden*, 1996.

[40] Pisinger, D., "Heuristics for the container loading problem," *European journal of operational research*, vol. 141, no. 2, pp. 382–392, 2002.

[41] Jin, Z., Ohno, K., and Du, J., "An efficient approach for the three-dimensional container packing problem with practical constraints," *Asia-Pacific Journal of Operational Research*, vol. 21, no. 03, pp. 279–295, 2004.

[42] Lin, J.-L., Chang, C.-H., and Yang, J.-Y., "A study of optimal system for multiple-constraint multiple-container packing problems," in *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*, Springer, 2006, pp. 1200–1210.

[43] Ceschia, S. and Schaerf, A., "Local search for a multi-drop multi-container loading problem," *Journal of Heuristics*, vol. 19, no. 2, pp. 275–294, 2013.

[44] Alonso, M., Alvarez-Valdes, R., Iori, M., and Parreño, F., "Mathematical models for multi container loading problems with practical constraints," *Computers & Industrial Engineering*, vol. 127, pp. 722–733, 2019.

[45] Nascimento, O. X. do, Queiroz, T. A. de, and Junqueira, L., "Practical constraints in the container loading problem: Comprehensive formulations and exact algorithm," *Computers & Operations Research*, vol. 128, p. 105 186, 2021.

[46] Dotoli, M., Epicoco, N., Falagario, M., Seatzu, C., and Turchiano, B., "A decision support system for optimizing operations at intermodal railroad terminals," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 3, pp. 487–501, 2016.

[47] Dotoli, M. and Epicoco, N., "A technique for the optimal management of containers' drayage at intermodal terminals," in *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, IEEE, 2016, pp. 000 566–000 571.

[48] Junqueira, L., Morabito, R., and Yamashita, D. S., "Mip-based approaches for the container loading problem with multi-drop constraints," *Annals of Operations Research*, vol. 199, no. 1, pp. 51–75, 2012.

[49] Lai, K., Xue, J., and Xu, B., "Container packing in a multi-customer delivering operation," *Computers & industrial engineering*, vol. 35, no. 1-2, pp. 323–326, 1998.

[50] Christensen, S. G. and Rousøe, D. M., "Container loading with multi-drop constraints," *International Transactions in Operational Research*, vol. 16, no. 6, pp. 727–743, 2009.

[51] Gendreau, M., Iori, M., Laporte, G., and Martello, S., "A tabu search algorithm for a routing and container loading problem," *Transportation Science*, vol. 40, no. 3, pp. 342–350, 2006.

[52] Tarantilis, C. D., Zachariadis, E. E., and Kiranoudis, C. T., "A hybrid metaheuristic algorithm for the integrated vehicle routing and three-dimensional container-loading problem," *IEEE Transactions on Intelligent Transportation Systems*, vol. 10, no. 2, pp. 255–271, 2009.

[53] Fuellerer, G., Doerner, K. F., Hartl, R. F., and Iori, M., "Metaheuristics for vehicle routing problems with three-dimensional loading constraints," *European Journal of Operational Research*, vol. 201, no. 3, pp. 751–759, 2010.

[54] Castellucci, P. B., Toledo, F. M., and Costa, A. M., "Output maximization container loading problem with time availability constraints," *Operations Research Perspectives*, vol. 6, p. 100 126, 2019.

[55] Xiang, X., Yu, C., Xu, H., and Zhu, S. X., "Optimization of heterogeneous container loading problem with adaptive genetic algorithm," *Complexity*, vol. 2018, 2018.

[56] Layeb, S. B., Jabloun, O., and Jaoua, A., "New heuristic for the single container loading problem," *International Journal of Economics & Strategic Management of Business Process*, vol. 8, pp. 1–7, 2017.

[57] Goel, R. and Maini, R., "Vehicle routing problem and its solution methodologies: A survey," *International Journal of Logistics Systems and Management*, vol. 28, no. 4, pp. 419–435, 2017.

[58] Dantzig, G. B. and Ramser, J. H., "The truck dispatching problem," *Management science*, vol. 6, no. 1, pp. 80–91, 1959.

[59] Golden, B. L., Magnanti, T. L., and Nguyen, H. Q., "Implementing vehicle routing algorithms," MASSACHUSETTS INST OF TECH CAMBRIDGE OPERATIONS RESEARCH CENTER, Tech. Rep., 1975.

[60] Bai, R., Chen, X., Chen, Z., *et al.*, "Analytics and machine learning in vehicle routing research," *CoRR*, vol. abs/2102.10012, 2021.

[61] Moura, A., "A multi-objective genetic algorithm for the vehicle routing with time windows and loading problem," in *Intelligent decision support*, Springer, 2008, pp. 187–201.

[62] Moura, A. and Oliveira, J. F., "An integrated approach to the vehicle routing and container loading problems," *OR spectrum*, vol. 31, no. 4, pp. 775–800, 2009.

[63] Rojas-Cuevas, I.-D., Caballero-Moralesb, S.-O., Sánchez-Partidab, D., and Martínez-Floresb, J.-L., "Three-axes rotation algorithm for the relaxed 3l-cvrp," *Jurnal Kejuruteraan*, vol. 33, no. 1, pp. 63–72, 2021.

[64] Rajaei, M., Moslehi, G., and Reisi-Nafchi, M., "The split heterogeneous vehicle routing problem with three-dimensional loading constraints on a large scale," *European Journal of Operational Research*, vol. 299, no. 2, pp. 706–721, 2022.

[65] Krebs, C., Ehmke, J. F., and Koch, H., "Effective loading in combined vehicle routing and container loading problems," *Computers & Operations Research*, vol. 149, p. 105 988, 2023.

[66] Küçük, M. and Yildiz, S. T., "Constraint programming-based solution approaches for three-dimensional loading capacitated vehicle routing problems," *Computers & Industrial Engineering*, vol. 171, p. 108 505, 2022.

[67] Meliani, Y., Hani, Y., Elhaq, S. L., and El Mhamedi, A., "A tabu search based approach for the heterogeneous fleet vehicle routing problem with three-dimensional loading constraints," *Applied Soft Computing*, vol. 126, p. 109 239, 2022.

[68] Wang, Y., Wei, Y., Wang, X., Wang, Z., and Wang, H., "A clustering-based extended genetic algorithm for the multidepot vehicle routing problem with time windows and three-dimensional loading constraints," *Applied Soft Computing*, vol. 133, p. 109 922, 2023.

[69] Psaraftis, H. N., "Dynamic vehicle routing problems," *Vehicle routing: Methods and studies*, vol. 16, pp. 223–248, 1988.

[70] Psaraftis, H. N., Wen, M., and Kontovas, C. A., "Dynamic vehicle routing problems: Three decades and counting," *Networks*, vol. 67, no. 1, pp. 3–31, 2016.

[71] Pillac, V., Gendreau, M., Guéret, C., and Medaglia, A. L., "A review of dynamic vehicle routing problems," *European Journal of Operational Research*, vol. 225, no. 1, pp. 1–11, 2013.

[72] Lorini, S., Potvin, J.-Y., and Zufferey, N., "Online vehicle routing and scheduling with dynamic travel times," *Computers & Operations Research*, vol. 38, no. 7, pp. 1086–1090, 2011.

[73] Li, J.-Q., Mirchandani, P. B., and Borenstein, D., "Real-time vehicle rerouting problems with time windows," *European Journal of Operational Research*, vol. 194, no. 3, pp. 711–727, 2009.

[74] Hu, X., Sun, L., and Liu, L., "A pam approach to handling disruptions in real-time vehicle routing problems," *Decision Support Systems*, vol. 54, no. 3, pp. 1380–1393, 2013.

[75] Xiang, X., Qiu, J., Xiao, J., and Zhang, X., "Demand coverage diversity based ant colony optimization for dynamic vehicle routing problems," *Engineering Applications of Artificial Intelligence*, vol. 91, p. 103 582, 2020.

[76] Okulewicz, M. and Mańdziuk, J., "A metaheuristic approach to solve dynamic vehicle routing problem in continuous search space," *Swarm and Evolutionary Computation*, vol. 48, pp. 44–61, 2019.

[77] Frohner, N., Horn, M., and Raidl, G. R., "Route duration prediction in a stochastic and dynamic vehicle routing problem with short delivery deadlines," *Procedia Computer Science*, vol. 180, pp. 366–370, 2021.

[78] Ng, K., Lee, C. K., Zhang, S., Wu, K., and Ho, W., "A multiple colonies artificial bee colony algorithm for a capacitated vehicle routing problem and re-routing strategies under time-dependent traffic congestion," *Computers & Industrial Engineering*, vol. 109, pp. 151–168, 2017.

[79] Sabar, N. R., Bhaskar, A., Chung, E., Turky, A., and Song, A., "A self-adaptive evolutionary algorithm for dynamic vehicle routing problems with traffic congestion," *Swarm and evolutionary computation*, vol. 44, pp. 1018–1027, 2019.

[80] Rios, B. H. O., Xavier, E. C., Miyazawa, F. K., Amorim, P., Curcio, E., and Santos, M. J., "Recent dynamic vehicle routing problems: A survey," *Computers & Industrial Engineering*, vol. 160, p. 107 604, 2021.

[81] Zhang, H., Ge, H., Yang, J., and Tong, Y., "Review of vehicle routing problems: Models, classification and solving algorithms," *Archives of Computational Methods in Engineering*, pp. 1–27, 2021.

[82] Moshref-Javadi, M., Hemmati, A., and Winkenbach, M., "A truck and drones model for last-mile delivery: A mathematical model and heuristic approach," *Applied Mathematical Modelling*, vol. 80, pp. 290–318, 2020. DOI: https://doi.org/10.1016/j.apm.2019.11.020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0307904X19306936.

[83] Chung, S. H., Sah, B., and Lee, J., "Optimization for drone and drone-truck combined operations: A review of the state of the art and future directions," *Computers & Operations Research*, vol. 123, p. 105 004, 2020. DOI: https://doi.org/10.1016/j.cor.2020.105004. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0305054820301210.

[84] Madani, B. and Ndiaye, M., "Hybrid truck-drone delivery systems: A systematic literature review," *IEEE Access*, 2022.

[85] Weng, Y.-Y., Wu, R.-Y., and Zheng, Y.-J., "Cooperative truck–drone delivery path optimization under urban traffic restriction," *Drones*, vol. 7, no. 1, p. 59, 2023.

[86] Wang, D., Hu, P., Du, J., Zhou, P., Deng, T., and Hu, M., "Routing and scheduling for hybrid truck-drone collaborative parcel delivery with independent and truck-carried drones," *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 10 483–10 495, 2019.

[87] Paris, A., Lopez, B. T., and How, J. P., "Dynamic landing of an autonomous quadrotor on a moving platform in turbulent wind conditions," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2020, pp. 9577–9583.

[88] Falanga, D., Zanchettin, A., Simovic, A., Delmerico, J., and Scaramuzza, D., "Vision-based autonomous quadrotor landing on a moving platform," in *2017 IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*, IEEE, 2017, pp. 200–207.

[89] Prayitno, A., Indrawati, V., and Trusulaw, I. I., "Fuzzy gain scheduling pid control for position of the ar. drone," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 8, no. 4, pp. 1939–1946, 2018.

[90] Hernandez, A., Copot, C., De Keyser, R., Vlas, T., and Nascu, I., "Identification and path following control of an ar. drone quadrotor," in *2013 17th international conference on system theory, control and computing (ICSTCC)*, IEEE, 2013, pp. 583–588.

[91] Richards, A. and How, J., "Decentralized model predictive control of cooperating uavs," in *2004 43rd IEEE Conference on Decision and Control (CDC)(IEEE Cat. No. 04CH37601)*, IEEE, vol. 4, 2004, pp. 4286–4291.

[92] Feng, Y., Zhang, C., Baek, S., Rawashdeh, S., and Mohammadi, A., "Autonomous landing of a uav on a moving platform using model predictive control," *Drones*, vol. 2, no. 4, p. 34, 2018.

[93] Setyawan, G. E., Kurniawan, W., and Gaol, A. C. L., "Linear quadratic regulator controller (lqr) for ar. drone's safe landing," in *2019 International Conference on Sustainable Information Engineering and Technology (SIET)*, 2019, pp. 228–233. DOI: 10.1109/SIET48054.2019.8986078.

[94] Dentler, J., Kannan, S., Mendez, M. A. O., and Voos, H., "A real-time model predictive position control with collision avoidance for commercial low-cost quadrotors," in *2016 IEEE Conference on Control Applications (CCA)*, 2016, pp. 519–525. DOI: 10.1109/CCA.2016.7587882.

[95] Keviczky, T., Borrelli, F., Fregene, K., Godbole, D., and Balas, G. J., "Decentralized receding horizon control and coordination of autonomous vehicle formations," *IEEE Transactions on Control Systems Technology*, vol. 16, no. 1, pp. 19–33, 2008. DOI: 10.1109/TCST.2007.903066.

Part 1: The Automated Storage Systems

Chapter 3

Automating Bin Packing: a Layer Building Matheuristics for Cost Effective Logistics

III

Abstract

This chapter addresses the problem of automating the definition of feasible pallets' configurations. This issue is crucial for the competitiveness of logistics companies and is still one of the most difficult problems in internal logistics. In fact, it requires the fast solution of a three-dimensional Bin Packing Problem (3D-BPP) with additional logistic specifications that are fundamental in real applications. To this aim, this chapter proposes a matheuristics that, given a set of items, provides feasible pallets configurations that satisfy the practical requirements of items' grouping by logistic features, load bearing, stability, height homogeneity, overhang as well as weight limits, and robotized layer picking. The proposed matheuristics combines a mixed integer linear programming (MILP) formulation of the 3D-Single Bin-Size BPP (3D-SBSBPP) and a layer building heuristics. In particular, the feasible pallets configurations are obtained by sequentially solving two MILP sub-problems: the first, given the set of items to be packed, aims at minimizing the unused space in each layer and thus the number of layers; the latter aims at minimizing the number of shipping bins given the set of layers obtained from the first problem. The approach is extensively tested and compared with existing approaches. For its validation both realistic data-sets drawn from the literature and real data-sets, obtained from the Italian logistics leader E80 Group, are considered. The resulting outcomes show the effectiveness of the method in providing high-quality bin configurations in short computational times.

Contents

3.1 Introduction

The industrial sector is experiencing the so-called fourth industrial revolution, based on the pervasive use of the enabling technologies of *Industry 4.0* [1] to increase productivity and quality standards of production processes [2]. In this context, the novel concept of *Logistics 4.0* has been introduced, aiming at adapting the general *Industry 4.0* pillars to logistics (e.g., automated warehouse, digitization of freight and information flows, and transport tracing) [3]–[8].

Among the various internal logistic activities, the packing of items on pallets is one of the most complex and resource-consuming. It requires the resolution of the three-dimensional Bin

Packing Problem (3D-BPP) with additional logistic specifications that ensure the stability and safety of the cargo. Its automatic solution can allow the warehouse manager to rapidly obtain feasible configurations of pallets with respect to specific cost functions and packing constraints, to reduce the number of composed pallets and the time required for their packing, and to increase the throughput of the company. Although commercial Warehouse Management Systems (WMSs) offer a large variety of functionalities for the management of internal logistics, effective algorithms that automate the definition of feasible pallets' configurations are still lacking. In particular, the available commercial solutions mainly offer 3D graphical tools that allow simulating the manual composition of items into bins from a visual perspective only. In addition, in the related literature (as detailed in Section 2.1.1 of Chapter 2.3) contributions to the definition and resolution of the 3D-BPP for internal logistic applications are limited, especially when compared with their two-dimensional counterparts, and they only include a restricted set of constraints that partially represent the business rules adopted in real logistic systems. Furthermore, the few existing literature contributions mainly address the 3D-BPP for simplified simulation scenarios and a recent survey highlights the lack of realistic benchmark data-sets [9]. With the aim of facilitating the transition from Logistics 3.0 to Logistics 4.0, in this chapter it is proposed a matheuristic algorithm [10] that can be integrated in the WMS to automatically and efficiently provide feasible pallets configurations. The proposed matheuristics allows defining feasible pallets configurations while taking into account the practical requirements of items' grouping by logistic features, load bearing, stability, height homogeneity, overhang as well as weight limits, and robotized layer picking. The method consists in a layer building heuristics that is based on the sequential solution of two Mixed Integer Linear Programming (MILP) sub-problems: the former, given the set of items to be packed, aims at minimizing the unused space in each layer and thus the number of layers; the latter aims at minimizing the number of shipping bins given the set of layers obtained from the first problem resolution.

It is important to highlight that the proposed matheuristics allows packing homogeneous and heterogeneous items into multiple single-sized bins and properly including particular logistic business rules, such as the *robotized layer picking* (i.e., the picking of entire layers composed by only one type of item by robotic manipulators) and the grouping of items by specific logistic features (i.e., delivery number, identification number, and category of product). The matheuristics is extensively tested to evaluate its performance in terms of quality of results and computational efficiency. Specifically, the first set of tests compares the outcomes and computational time of the 3D-Single Bin-Size BPP (3D-SBSBPP) with the ones of the proposed algorithm including only geometric, weight, and overhang constraints. The second set of tests provides a comparison between this layer-building matheuristics and the literature algorithm proposed in [11], which is one of the most recent and most complete matheuristics in terms of logistic requirements. The third regards the performance analysis of the layer-building algorithm with a real data-set obtained by an Italian industry leader of the logistic sector. The obtained outcomes show the ability of the method to provide high quality results in short computational times. As it emerges from the discussion of the literature review, only a few contributions address the 3D-SBSBPP from the general perspective of the real logistic sector and present applications to real case studies.

In this chapter, it is proposed a matheuristics of practical applicability for the automatic and efficient resolution of the 3D-SBSBPP. In particular, it is defined a novel MILP-based layer building algorithm that efficiently determines feasible pallet configurations, minimizing the unused space and fulfilling a set of geometric and logistic specifications, i.e., items' grouping by logistic features, load bearing, stability, height homogeneity, overhang and weight limits, and robotized layer picking. In contrast with the contributions analyzed in Section 2.1.1, the proposed algorithm allows the automatic management of the packing process, starting from the analysis of the shipment list and allowing the definition of the most suitable configuration of bins to be delivered

to the final customers in a short computational time. A synthetic comparison of this approach with the related literature is reported in Table 2.1. In particular, differently from [12] and [13], it allows the definition of multiple bins of homogeneous and heterogeneous items. With respect to [14] and [15], this work takes into account also compatibility constraints among items, e.g., food is not packed with chemical products, and fragility constraints. Moreover, similarly to [16], here it is considered the decomposition of the problem into the two Layer Building and Bin Building phases, but it as item support, bin stability, and load bearing for the application of the method in the logistic field. The contributions of this chapter can be then summarized as follows:

- the formulation of the 3D-SBSBPP as a MILP-based layer building BPP that first optimizes the layers composition and then the bin configuration. It includes not only geometric constraints, but also logistic requirements, i.e., load bearing, stability, height homogeneity, overhang and weight limits (see Table 2.1);
- the definion of a three-phase matheuristics based on the layer building formulation of the 3D-SBSBPP that allows, in a short computational time, the proper and automatic management of the packing process from the analysis of the shipment list to the definition of the most suitable configuration of bins to be delivered to the final customer. The method allows fulfilling further logistic constraints that are not included in the mathematical formulation of the problem, i.e, the grouping of items by logistic features, such as, delivery number, identification number, and category of product, and the robotized layer picking function (see Table 2.1);
- the extensive computational tests campaign to compare the performance of the proposed algorithm with respect to a reference method using both realistic data-sets drawn from the literature and real data-sets. In particular, differently from related works that consider testing instances with geometrical features only, it shows the effectiveness of the proposed methodology on industry-size scenarios including practical constraints required by logistic companies.

3.2 The 3D-SBSBPP Formulation

In this section it is presented the mathematical models on which the proposed matheuristics relies, which is detailed in Section 3.3. In particular, the logistic 3D-SBSBPP is formulated with two sub-problems based on [17], namely:

- Layer building sub-problem: given the set of items to be packed, this sub-problem aims at minimizing the unused space in each layer and thus the number of items' layers, while fulfilling geometric constraints.
- Bin building sub-problem: given the set of layers obtained from the Layer Building sub-problem, this sub-problem aims at minimizing the number of shipping bins, while fulfilling geometric and safety constraints.

Table 3.2 shows all the parameters used in the proposed formulation, while the variables are presented in Table 3.1.

3.2.1 Layer Building Sub-problem

In this sub-problem the goal is to arrange the given items into a minimum number V of layers having the highest fill ratio and including items with similar heights. The layers are composed in accordance with an iterative procedure. Initially, the first layer is composed extracting the

Table 3.1: List of variables for the layer building an bin building sub-problems

Name	Description	Value
Layer Building		
i, i'	indices of items	$[1, \mathrm{N}_j]$
p_i	binary variable indicating if item i is inside (1) or outside (0) the layer	{0,1}
x_i	x-axis coordinate of the left-bottom corner of item i in the layer	$\mathbb{R}^+$
y_i	y-axis coordinate of the left-bottom corner of item i in the layer	$\mathbb{R}^+$
$l_{\mathrm{x}i}$	binary variable indicating whether the length of item i is parallel to the x-axis (1) or not (0)	{0,1}
$l_{\mathrm{y}i}$	binary variable indicating whether the length of item i is parallel to the y-axis (1) or not (0)	{0,1}
$le_{(i,i')}$	binary variable indicating whether item i is on the left (1) of item i' or not (0)	{0,1}
$r_{(i,i')}$	binary variable indicating whether item i is on the right (1) of item i' or not (0)	{0,1}
$f_{(i,i')}$	binary variable indicating whether item i is in front (1) of item i' or not (0)	{0,1}
$b_{(i,i')}$	binary variable indicating whether item i is behind (1) item i' or not (0)	{0,1}
Bin Building		
j, j'	indices of layers	$[1, \mathrm{V}]$
k	index of bin	$[1, \mathrm{M}_{\max}]$
n_k	binary variable indicating whether bin k is empty (0) or not (1)	{0,1}
$v_{j,k}$	binary variable indicating whether layer j is inside (1) or outside (0) bin k	{0,1}
z_j	z-axis coordinate of the left-bottom corner of layer j	$\mathbb{R}^+$
$o_{(j,j')}$	binary variable indicating whether layer j is above (1) layer j' or not (0)	{0,1}
$u_{(j,j')}$	binary variable indicating whether layer j is below (1) layer j' or not (0)	{0,1}

optimal subset from all the N items, which maximizes the fill ratio while fulfilling geometric constraints, overhang limits and height homogeneity requirements, i.e., ensuring that the height difference between the selected items is lower than a given threshold. For the second layer, the optimal selection procedure is applied to the remaining available items. The optimization steps are then iterated until all items are paired with a layer. For the composition of a generic layer j, given the set $\mathcal{n}_j \subseteq \mathcal{n}$ of N_j available items, the layer building model is formulated as follows:

$$\min \left((\Theta + \mathrm{O})(\Lambda + \mathrm{Q}) - \sum_{i \in \mathcal{n}_j} p_i \theta_i \lambda_i \right) \tag{3.1}$$

subject to:

$$x_i + \theta_i l_{\mathrm{x}i} + \lambda_i (1 - l_{\mathrm{x}i}) \leq \Theta + \mathrm{O} + (1 - p_i)\mathrm{L}, \forall i \tag{3.2}$$

$$y_i + \lambda_i (1 - l_{\mathrm{y}i}) + \theta_i l_{\mathrm{y}i} \leq \Lambda + \mathrm{Q} + (1 - p_i)\mathrm{L}, \forall i \tag{3.3}$$

$$le_{(i,i')} + r_{(i,i')} + b_{(i,i')} + f_{(i,i')} \geq p_i + p_{i'} - 1, \forall i, i', i' < i \tag{3.4}$$

$$x_i + \theta_i l_{\mathrm{x}i} + \lambda_i (1 - l_{\mathrm{x}i}) \leq x_{i'} + (1 - le_{(i,i')})\mathrm{L}, \forall i, i', i' < i \tag{3.5}$$

$$x_{i'} + \theta_{i'} l_{\mathrm{x}i'} + \lambda_{i'} (1 - l_{\mathrm{x}i'}) \leq x_i + (1 - r_{(i,i')})\mathrm{L}, \forall i, i', i' < i \tag{3.6}$$

$$y_i + \lambda_i (1 - l_{\mathrm{y}i}) + \theta_i l_{\mathrm{y}i} \leq y_{i'} + (1 - b_{(i,i')})\mathrm{L}, \forall i', i' < i \tag{3.7}$$

$$y_{i'} + \lambda_{i'} (1 - l_{\mathrm{y}i'}) + \theta_{i'} l_{\mathrm{y}i'} \leq y_i + (1 - f_{(i,i')})\mathrm{L}, \forall i, i', i' < i \tag{3.8}$$

$$(\psi_i - \psi_{i'}) \left(le_{(i,i')} + r_{(i,i')} + f_{(i,i')} + b_{(i,i')} \right) \leq \mathrm{G}, \forall i, i', i' < i \tag{3.9}$$

$$(\psi_i - \psi_{i'}) \left(le_{(i,i')} + r_{(i,i')} + f_{(i,i')} + b_{(i,i')} \right) \geq \text{-G}, \forall i, i', i' < i \tag{3.10}$$

$$l_{\mathrm{x}i} + l_{\mathrm{y}i} = 1, \forall i \tag{3.11}$$

$$0 \leq x_i \leq \Theta, \forall i \tag{3.12}$$

$$0 \leq y_i \leq \Lambda, \forall i \tag{3.13}$$

$$le_{(i,i')}, r_{(i,i')}, b_{(i,i')}, f_{(i,i')} \in \{0, 1\}, \forall i, i', i' < i \tag{3.14}$$

$$p_i \in \{0, 1\}, \forall i \tag{3.15}$$

$$l_{\mathrm{x}i}, l_{\mathrm{y}i} \in \{0, 1\}, \forall i. \tag{3.16}$$

Table 3.2: List of parameters for the layer building and bin building sub-problems

Name	Description
	Common
L	an arbitrary large number
s_i	stability index of item i
c_i	maximum load supported by an item i
γ_S	weight factor to estimate the stability of layer j
γ_C	weight factor to estimate the maximum load supported by a layer j
H_j	height of layer j
A_j	area of layer j
S_j	stability index of layer j
W_j	weight of layer j
C_j	maximum load supported by layer j
	Layer Building
M_{max}	maximum number of available bins to be composed
N	total number of items
Θ	width of pallets
Λ	length of pallets
Ψ	maximum height of bins
O	maximum width overhang
Q	maximum length overhang
θ_i	width of item i
λ_i	length of item i
ψ_i	height of item i
ω_i	weight of item i
	Bin Building
F	maximum load supported by a pallet
V	total number of layers
G	maximum height gap among items of the same layer
B	maximum area gap among two consecutive layers

The objective in (3.1) is to maximize the fill ratio of the layer (i.e., to minimize the horizontal area of the pallet that is not occupied by selected items). Constraints (3.2)-(3.3) guarantee that each item is contained in the dimensions of the layer allowing a overhang tolerance for the x and y axis; moreover, they allow the rotation of the item by 90 degrees along the vertical axis. Constraints (3.4)-(3.8) ensure the assignment of the relative position of two items without overlapping, allowing the combination of the positions front, back, left, and right. Constraints (3.9)-(3.10) ensure that the maximum gap between the items height inside one layer is lower than a threshold, thus keeping layers as homogeneous as possible and contributing to the stability of the overall configuration. Constraints (3.11) guarantee the unique assignment of the orientation of each item i. Finally, constraints (3.12)-(3.13) and (3.14)-(3.16) specify the bounding and the integrality conditions on the defined real and binary decision variables, respectively.

Summing up, the resulting MILP problem (3.1)-(3.16) consists in determining the $2N_j$ real and $N_j(2N_j + 1)$ binary variables characterizing the layers and listed in the first part of Table 3.1, which minimize the objective function in (3.1) and meet the N_j equality constraints (3.11), the

$\frac{7}{2}\mathrm{N}_j(\mathrm{N}_j+3)$ inequality constraints (3.2)-(3.10), the $4\mathrm{N}_j$ bounding constraints (3.12)-(3.13), and the $\mathrm{N}_j(2\mathrm{N}_j+1)$ integrality constraints in (3.14)-(3.16).

The iterative resolution of (3.1)-(3.16) allows determining the composition of layers (the items allocated to each layer and their location in terms of coordinates and vertical rotation) and, after executing the above MILP problem, the parameters representing the physical features of layers are then determined as follows:

$$H_j = \max_{i \in n_j} \psi_i, \forall j \tag{3.17}$$

$$A_j = \sum_{i \in n_j} \lambda_i \theta_i, \forall j \tag{3.18}$$

$$S_j = \gamma_S \sum_{i \in n_j} \mathrm{s}_i, \forall j \tag{3.19}$$

$$C_j = \gamma_C \sum_{i \in n_j} \mathrm{c}_i, \forall j \tag{3.20}$$

$$W_j = \sum_{i \in n_j} \omega_i, \forall j. \tag{3.21}$$

In particular, the height of a given layer is set equal to the height of the highest selected item; the layer occupied area is the sum of the areas occupied by all the selected items; the layer stability is defined as the sum (scaled by factor γ_S) of the values associated to the selected items; the maximum load supported by the given layer is estimated as the sum (scaled by factor γ_C) of the values associated to the selected items; the weight of the layer is given by the sum of the weights of all the items contained in it.

3.2.2 Bin Building Sub-problem

In this sub-problem, given the set of the V layers obtained by solving the optimization problem (3.1)-(3.16) and the corresponding parameters computed by (3.17)-(3.21), the aim is to minimize the number of bins composed by the given layers, while fulfilling geometric and safety requirements.

The bin building problem is formulated as follows:

$$\min \sum_{k=1}^{\mathrm{M_{max}}} n_k \tag{3.22}$$

subject to:

$$\sum_{j=1}^{\mathrm{V}} v_{j,k} \leq \mathrm{V}\,\mathrm{n}_k, \forall k \tag{3.23}$$

$$\sum_{k=1}^{\mathrm{M_{max}}} v_{j,k} = 1, \forall j \tag{3.24}$$

$$z_j + H_j \leq \Psi + (1 - v_{j,k})\mathrm{L}, \forall k, j \tag{3.25}$$

$$o_{(j,j')} + u_{(j,j')} \geq v_{j,k} + v_{j',k} - 1, \forall k, j, j', j' < j \tag{3.26}$$

$$z_j + \psi_j \leq z_{j'} + (1 - o_{(j,j')})\mathrm{L}, \forall j, j', j' < j \tag{3.27}$$

$$z_{j'} + \psi_{j'} \leq z_j + (1 - u_{(j,j')})\mathrm{L}, \forall j, j', j' < j \tag{3.28}$$

$$\sum_{j'=1}^{\mathrm{V}} W_{j'} o_{(j',j)} \leq C_j, \forall j \tag{3.29}$$

$$\sum_{j=1}^{\mathrm{V}} v_{j,k} W_j \leq \mathrm{F}, \forall k \tag{3.30}$$

$$S_j o_{(j,j')} \leq S_{j'} u_{(j',j)}, \forall j, j', j' < j \tag{3.31}$$

$$S_j u_{(j,j')} \geq S_{j'} o_{(j',j)}, \forall j, j', j' < j \tag{3.32}$$

$$(A_j - A_{j'}) o_{(j,j')} \leq \mathrm{B}, \forall j, j', j' < j \tag{3.33}$$

$$(A_j - A_{j'}) u_{(j,j')} \leq \mathrm{B}, \forall j, j', j' < j \tag{3.34}$$

$$0 \leq z_j \leq \Psi, \forall j \tag{3.35}$$

$$n_k \geq n_{k+1}, \forall k \in \{1, \ldots, \mathrm{M_{max}} - 1\} \tag{3.36}$$

$$o_{(j,j')}, u_{(j,j')} \in \{0, 1\}, \forall j, j', j' < j \tag{3.37}$$

$$v_{j,k} \in \{0, 1\}, \forall k, j \tag{3.38}$$

$$n_k \in \{0, 1\}, \forall k. \tag{3.39}$$

The objective in (3.22) is to minimize the number of bins to be composed, given the set of layers. Constraints (3.23) ensure the consistency between binary variables $v_{j,k}$ ($\forall j$) and n_k for each bin k, i.e., if any layer is assigned to a bin, the bin is considered not empty. Constraints (3.24) make sure that each layer can be part at most of one bin. Constraints (3.25) guarantee that the placement of each layer does not exceed the maximum height of the bin. Constraints (3.26) concern the relative position that two consecutive layers can assume inside the bin (i.e., on top or below). Constraints (3.27) - (3.28) are related to the non-overlapping of two layers placed in the same bin. Constraints (3.29) limit the maximum load that a single layer can withstand. Constraints (3.30) limit the maximum weight that a single pallet can withstand. The safety constraints (3.31) and (3.32) ensure that the stability index of each layer is higher than or equal to the stability index of the respective above layers, while (3.33) and (3.34) impose that the gap of area between two consecutive layers must not be greater than the given threshold B. Finally, constraints (3.35), (3.36) and (3.37)-(3.39) specify the bounding, the validity, and integrality conditions on the defined real and binary decision variables, respectively.

Summing up, the resulting MILP problem (3.22)-(3.39) consists in determining the V real and $2\mathrm{M_{max}}(1 + \mathrm{V}) + \mathrm{V}(\mathrm{V} - 1)$ binary variables characterizing the bins and listed in the second part of Table 3.1, which minimize the objective function in (3.22) and meet the V equality constraints (3.24), the $\mathrm{M_{max}}(2 + \frac{1}{2}\mathrm{V}(\mathrm{V} + 1)) + \mathrm{V}(2 + 3(\mathrm{V} - 1))$ inequality constraints (3.23) and (3.26)-(3.39), the 2V bounding constraints (3.35), the $\frac{1}{2}(\mathrm{M_{max}}(\mathrm{M_{max}} - 1))$ validity constraints (3.36), and the $\mathrm{M_{max}}(1 + \mathrm{V}) + \mathrm{V}(\mathrm{V} - 1)$ integrality constraints (3.37)-(3.39).

3.3 The Matheuristics for the Automated 3D-SBSBPP

In this section, it is described the proposed matheuristics for the logistic 3D-SBSBPP, which is based on the mathematical models described in section 3.2. It is worth highlighting that, the algorithm takes as input a shipment list that includes items with different IDs and different categories, to be delivered to different clients. Thus, the algorithm is in charge of creating monoitem, monocategory, and mixed layers. As shown in Figure 3.1, the proposed matheuristics is composed by 3 different phases, namely, Grouping, Layer Building, and Bin Building, which are represented in the flowchart by dashed boxes and are executed sequentially.

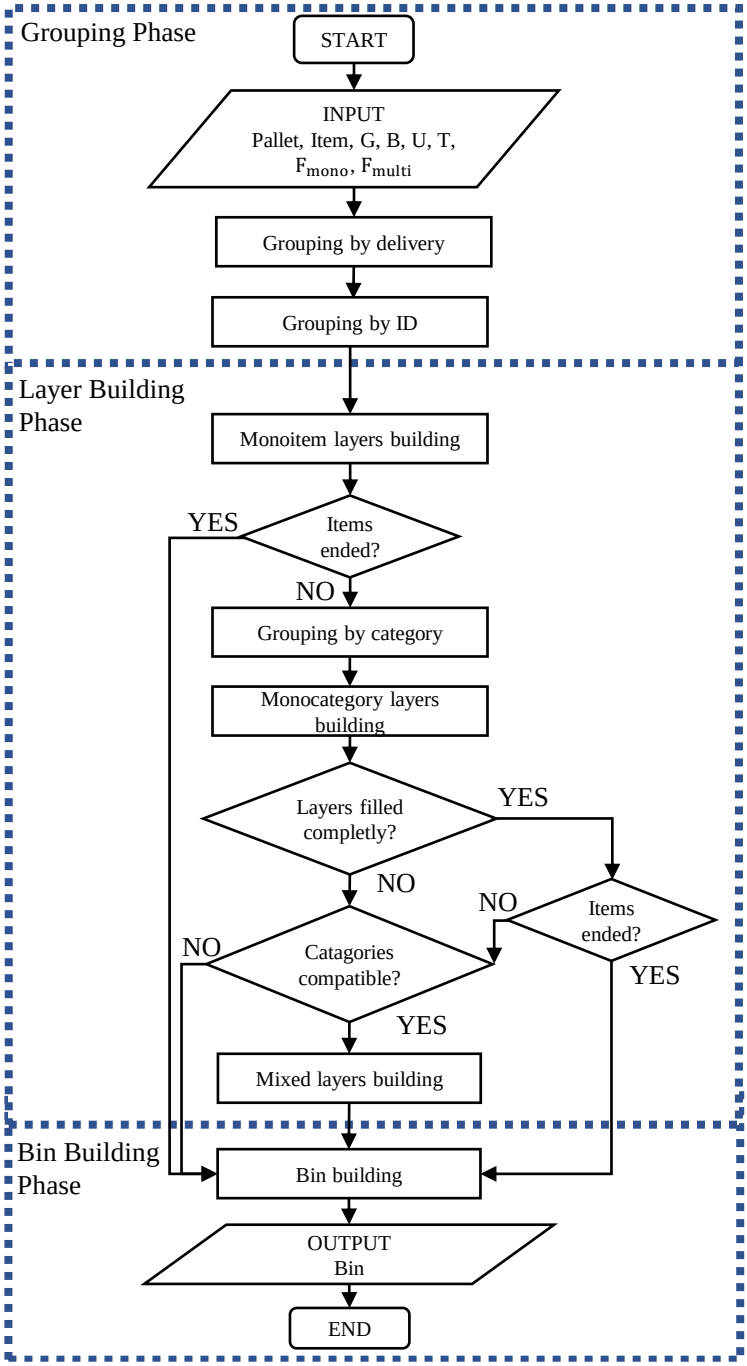


Figure 3.1: High-level flow chart of the proposed matheuristics.

Grouping: this is a pre-processing phase aimed mainly at grouping items by delivery and ID. This procedure receives as input the data related to a shipment including the list of deliveries associated to various customers. First, this phase initializes all the parameters of the problem,

i.e., the parameters related to items (N, θ_i, λ_i, ψ_i, s_i, a_i, ω_i), the class of parameters related to pallets (V, Θ, Λ, Ψ, O, Q), and the parameters related to the creation of layers (G, B, U, T, FR_{mono}, FR_{multi}), where U and T are the maximum allowable layer picking weight and height, and FR_{mono} and FR_{multi} are the minimum admitted values of the fill ratio of monoitem layers and of monocategory and mixed layers. The fill ratio indicates the percentage of the layers' area occupied by the respective items, while parameters O and Q are set according to the dimensions of the input items, ensuring that at least the half base of the smallest item is placed inside the bin. They are calculated as follows:

$$O = Q < \frac{\min_i \{\theta_i, \lambda_i\}}{2}. \tag{3.40}$$

Moreover, with the aim of facilitating the delivery and the loading/unloading of bins from transport units to customers, items belonging to a specific delivery are grouped, so that such items can be packed together inside the bins. Subsequently, given a delivery, the algorithm further groups the items by ID.

Layer Building: the main purpose of this phase is to obtain feasible configurations of monoitem layers, monocategory layers, and mixed layers. In particular, monoitem layers are created starting from each monoitem set by means of a heuristic procedure that creates strips of items inside of the layer by starting from its left bottom corner. A monoitem layer is created if the area of the items fully covers the area of the layer (i.e., the fill ratio of the j-th layer is $FR_j \geq FR_{mono}$) and also both the length and width of the items fit the length and width of the layer. In particular, the items are sequentially positioned in the layer by iteratively assigning the coordinates of the left bottom corner of each item. For the monoitem layers, the robotized layer picking is enabled only if the total weight and the height of the layer are lower than or equal to the maximum allowable layer picking weight (U) and maximum allowable layer picking height (T). If all the items of the shipment are inserted in monoitem layers, the algorithm executes the subsequent Bin Building phase, otherwise the remaining items are processed for creating monocategory and mixed layers. Specifically, the items are first grouped by category. For each category, the items are grouped in subsets $\mathcal{N}_j$ for which the summation of the items' base area satisfies the relation $FR_j \geq FR_{multi}$. The optimization problem (3.1)-(3.16) is then iteratively solved over the subsets $\mathcal{N}_j$, until all items are assigned to a layer. Note that at each step a MILP solver iterates until a solution is found or the computational time is greater than a given threshold Δ_t (i.e., an approximate solution is found). At each iteration, the obtained layer configuration is saved if it satisfies the requirement on the fill ratio, i.e., $FR_j \geq FR_{multi}$, otherwise the corresponding items are recombined with the remaining items and novel subsets $\mathcal{N}_j$, containing all the items that are still not assigned to any bin, are defined. The eventual residual items of different but compatible categories are combined in mixed layers following the same steps described for monocategory layers. In the eventuality of residual compatible items, further mixed layers will be composed by relaxing the height homogeneity and fill ratio requirements.

The algorithm then moves to the Bin Building phase in the following cases: (1) all the items included in the monocategory sets are assigned to layers (both monocategory and mixed layers); (2) not all items are assigned to a layer, but these residual items belong to incompatible categories (i.e., categories that cannot be combined in the same layer, e.g., food and chemical products).

Bin Building: this phase aims at properly combining the layers computed in the previous phases (i.e., monoitem, monocategory, and mixed layers) into a minimum number of bins. In particular, this procedure is executed on a delivery basis, i.e., all the layers related to a given delivery are grouped by compatible categories and used to solve the bin building problem (3.22)-(3.39) presented in Section 3.2.2.

It is important to notice that the proposed matheuristic algorithm provides feasible solutions in a short computational time. The quality of the results and the scalability of the method are obtained empirically. In the experimental results section, there is a discussion of both aspects and a comparison of the results of the presented algorithm with those of the exact resolution.

3.4 Experimental Results

This section is devoted to the validation of the performance of the proposed matheuristics. All tests are performed on a laptop equipped with a 2.20 GHz Intel Core i7-8750H CPU and 32 GB RAM using C# language [18], combined with the Glop linear solver [19]. In particular, due to the nature of the elements characterizing the 3D-SBSBPP, object-oriented programming has been employed, in order to properly represent the structure and relations of the problem data.

In the following sub-sections the proposed matheuristics is tested first on a small data-set where the results and performance of the algorithm are compared with the ones of the 3D-SBSBPP basic formulation presented in Section 3.4.1.1. Then the method is compared with a reference method using both realistic data-sets drawn from the literature and realistic industrial data-sets [11]. Finally, the algorithm is tested on real data provided by the Italian logistic company E80 Group [20] to further investigate its performance.

The following performance indicators are considered to analyse the performance of the algorithm.
Number of created layers and bins:

- M: total number of obtained bins;
- V: total number of obtained layers;

Fill ratio [%]:

- AvgFR_V: layers' average fill ratio;

$$\text{AvgFR}_\text{V} = \frac{100}{\text{V}} \sum_{j=1}^{\text{V}} (1 - \frac{(\Theta + \text{O})(\Lambda + \text{Q}) - \sum_{i=1}^{\text{N}_j} \text{a}_i}{(\Theta + \text{O})(\Lambda + \text{Q})})$$

where a_i is the base area of the $i-$th item obtained as $\theta_i \lambda_i$.

- AvgFR_M: bins' average fill ratio;

$$\text{AvgFR}_\text{M} = \frac{100}{\text{M}} \sum_{k=1}^{\text{M}} (1 - \frac{(\Theta + \text{O})(\Lambda + \text{Q})\Psi - \sum_{j=1}^{\text{V}_k} \sum_{i=1}^{\text{N}_{j,k}} \text{d}_i}{(\Theta + \text{O})(\Lambda + \text{Q})\Psi})$$

where d_i is the volume of the $i-$th item obtained as $\theta_i \lambda_i \psi_i$.
Computational time [s]:

- T_ex: total computational time of the algorithm;

Stability indices S1_k and S2_k, firstly qualitatively described by [21] and used also by [12], are formalized as follows:

- S1_k: average number of items positioned below each item, in case this is not positioned directly on the pallet, i.e., not considering the lowest layer, formulated as follows:

$$S1_k = \frac{1}{V_k - 1} \sum_{j=1}^{V_k - 1} \left(\frac{\sum_{i'=1}^{N_{j+1,k}} \sum_{i=1}^{N_{j,k}} \mu_{(i,i')}}{N_{j+1,k}} \right)$$

where $\mu_{(i,i')}$ is equal to 1 if item i is under item i' otherwise is equal to 0

- $S2_k$: average percentage of items which are not surrounded by other items in at least 3 sides inside of bin k, formulated as follows:

$$S2_k = \frac{100}{N_k} \sum_{j=1}^{V_k} \sum_{i=1}^{N_{j,k}} \min\Bigg\{1, \max\Bigg\{0, 3 - \Bigg(\min\Big\{1, \sum_{i'=1}^{N_{j,k}} b_{(i,i'),j}\Big\} + \min\Big\{1, \sum_{i'=1}^{N_{j,k}} f_{(i,i'),j}\Big\} + \min\Big\{1, \sum_{i'=1}^{N_{j,k}} le_{(i,i'),j}\Big\} + \min\Big\{1, \sum_{i'=1}^{N_{j,k}} r_{(i,i'),j}\Big\} \Bigg) \Bigg\} \Bigg\}$$

where $b_{(i,i'),j}$ is equal to 1 if item i is behind item i' in layer j otherwise is 0, $f_{(i,i'),j}$ is equal to 1 if item i is in front of item i' in layer j otherwise is 0, $le_{(i,i'),j}$ is equal to 1 if item i is on the left of item i' in layer j otherwise is 0, and $r_{(i,i'),j}$ is equal to 1 if item i is on the right of item i' in layer j otherwise is 0. As for the parameters, N_j is the number of items in layer j, V_k is the number of layers of bin k, and N_k is the number of items of bin k.

The overhang index is formulated as:

- $S3_k$: average value of the overhanging ratios over all the layers assigned to the $k-$th bin. For each layer the overhanging ratio is computed dividing the actual layer overhanging area by the maximum pallet overhanging area (i.e., $(\Theta + O)(\Lambda + Q) - \Theta\Lambda$).

In the set-up of each implemented scenario the parameters for the 3D-SBSBPP are set as follows:

- satisfactory monoitem layer fill ratio $FR_{mono} = 99\%$;
- satisfactory monocategory or mixed layer fill ratio $FR_{multi} = 90\%$;
- maximum height gap among items of the same layer G=20 mm;
- maximum area gap among two consecutive layers B=10000 mm^2;
- maximum load supported by the layer picking U=1000 Kg;
- maximum height supported by the layer picking T=1000 mm.

Finally, the only remaining set-up parameter for the proposed algorithm is the maximum execution time in which the solver of Layer Building Phase has to find the best configuration for the creation of each layer, that it is setted Δ_t=90 seconds.

3.4.1 Comparison with an Exact Method

In this subsection the proposed matheuristic algorithm is compared with the exact solution of the logistic 3D-SBSBPP. Note that, as highlighted in the comparative review in [9], only a few contributions are available in the literature that apply exact methods to the 3D-SBSBPP. This is mainly due to the difficulties in representing patterns or practical packing constraints. For this reason, in the following there is the report of the MILP formulation of the 3D-SBSBPP based on the literature formulation by Chen *et al.* [17] that includes geometric, overhang, rotation, and weight constraints and it is compared with the proposed approach including the same requirements.

3.4.1.1 The Exact 3D-SBSBPP Formulation

This MILP formulation of the logistic 3D-SBSBPP aims at minimizing the number of shipping bins used for packing a given set of items, while fulfilling a basic set of geometric and safety constraints. Differently from the classical 3D-BPP in [22], this formulation takes additional constraints into account. Specifically, a tolerance excess band (both in width and length) in the size of the bin base dynamically calculated according to the dimensions of the items (see Section 3.3), rotation along the z axis, and weight limits for the pallet. The notation and meaning of parameters and variables of the formulation are explained respectively in Tables 3.2 and 3.1.

The mathematical model is defined as follows:

$$\min \sum_{k=1}^{\mathrm{M_{max}}} (\Theta + \mathrm{O})(\Lambda + \mathrm{Q})\Psi n_k - \sum_{i=1}^{\mathrm{N}} \theta_i \lambda_i \psi_i \tag{3.41}$$

subject to:

$$\sum_{i=1}^{\mathrm{N}} p_{i,k} \leq \mathrm{N}\, \mathrm{n}_k \ , \forall k \tag{3.42}$$

$$\sum_{k=1}^{\mathrm{M_{max}}} p_{i,k} = 1, \forall i \tag{3.43}$$

$$\sum_{i=1}^{\mathrm{N}} p_{i,k}\, \omega_i \leq \mathrm{F}, \forall k \tag{3.44}$$

$$x_i + \theta_i l_{\mathrm{x}i} + \lambda_i (1 - l_{\mathrm{x}i}) \leq \Theta + \mathrm{O} + (1 - p_{i,k})\mathrm{L}, \forall k, i \tag{3.45}$$

$$y_i + \lambda_i (1 - l_{\mathrm{y}i}) + \theta_i l_{\mathrm{y}i} \leq \Lambda + \mathrm{Q} + (1 - p_{i,k})\mathrm{L}, \forall k, i \tag{3.46}$$

$$z_i + \psi_i \leq \Psi + (1 - p_{i,k})\mathrm{L}, \forall k, i \tag{3.47}$$

$$le_{(i,i')} + r_{(i,i')} + b_{(i,i')} + f_{(i,i')} + o_{(i,i')} + u_{(i,i')} \geq p_{i,k} + p_{i',k} - 1, \forall k, i, i', i' < i \tag{3.48}$$

$$x_i + \theta_i l_{\mathrm{x}i} + \lambda_i (1 - l_{\mathrm{x}i}) \leq x_{i'} + (1 - le_{(i,i')})\mathrm{L}, \forall i, i', i' < i \tag{3.49}$$

$$x_{i'} + \theta_{i'} l_{\mathrm{x}i'} + \lambda_{i'} (1 - l_{\mathrm{x}i'}) \leq x_i + (1 - r_{(i,i')})\mathrm{L}, \forall i, i', i' < i \tag{3.50}$$

$$y_i + \lambda_i (1 - l_{\mathrm{y}i}) + \theta_i l_{\mathrm{y}i} \leq y_{i'} + (1 - b_{(i,i')})\mathrm{L}, \forall i, i', i' < i \tag{3.51}$$

$$y_{i'} + \lambda_{i'} (1 - l_{\mathrm{y}i'}) + \theta_{i'} l_{\mathrm{y}i'} \leq y_i + (1 - f_{(i,i')})\mathrm{L}, \forall i, i', i' < i \tag{3.52}$$

$$z_i + \psi_i \leq z_{i'} + (1 - o_{(i,i')})\mathrm{L}, \forall i, i', i' < i \tag{3.53}$$

$$z_{i'} + \psi_{i'} \leq z_i + (1 - u_{(i,i')})\mathrm{L}, \forall i, i', i' < i \tag{3.54}$$

$$l_{xi} + l_{yi} = 1, \forall i \tag{3.55}$$
$$0 \leq x_i \leq \Theta, \forall i \tag{3.56}$$
$$0 \leq y_i \leq \Lambda, \forall i \tag{3.57}$$
$$0 \leq z_i \leq \Psi, \forall i \tag{3.58}$$
$$n_k \geq n_{k+1}, \forall k \in \{1, \ldots, M_{max-1}\} \tag{3.59}$$
$$n_k \in \{0, 1\}, \forall k \tag{3.60}$$
$$p_{i,k} \in \{0, 1\}, \forall i, k \tag{3.61}$$
$$le_{(i,i')}, r_{(i,i')}, b_{(i,i')} \in \{0, 1\}, \forall i, i', i' < i \tag{3.62}$$
$$f_{(i,i')}, o_{(i,i')}, u_{(i,i')} \in \{0, 1\}, \forall i, i', i' < i \tag{3.63}$$
$$l_{xi}, l_{yi} \in \{0, 1\}, \forall i. \tag{3.64}$$

The objective in (3.41) is the minimization of the unoccupied space over the total number of used bins. In addition, constraints (3.42) ensure the consistency between binary variables $p_{i,k}$ ($\forall i$) and n_k for each bin k, i.e., if any item is assigned to a bin, the bin is considered not empty. Constraints (3.43) make sure that each item can be part at most of one bin. Constraint (3.44) make sure that the overall weight of items allocated to each bin is not greater than the maximum weight supported by the pallet. Constraints (3.45)-(3.47) guarantee that each item is contained in the dimensions of the bin allowing a overhang tolerance for the x and y axis; moreover, they allow the rotation of the item by 90 degrees along the vertical axis. Constraints (3.48)-(3.54) are related to the relative positions that two items can assume inside the bin: (3.48) ensure the assignment of the relative position of two items allowing the combination of the positions front, back, left, right, over, and under, while (3.49)-(3.54) guarantee that those items do not overlap. Constraints (3.55) guarantee the unique assignment of the orientation of each item i. Finally, constraints (3.56)-(3.59) and (3.60)-(3.64) specify the bounding and integrality conditions on the defined real and binary decision variables, respectively.

Figure 3.2a shows an example of the relative position of two items (i and i') laying on the x/y plane, that is, item i is positioned on the front left side with respect to object i' in the same plane. Figure 3.2b represents the two different orientations that an item can assume with respect to the value of the variables l_{xi} and which may be 0 or 1 (for the sake of simplicity, the image is in 2D because the rotation is done on the x/y and so the height of the item is not influent).

Summing up, the resulting MILP problem (3.41)-(3.64) consists in determining 3N real and $M_{max}(N + 1) + N(3N - 1)$ binary variables, which minimize the objective function in (3.41) and meet the $\frac{M_{max}}{2}(4 + 7N + N^2) + 3N(N + 1)$ inequality constraints (3.42) and (3.44)-(3.54), the 2N equality constraints (3.43) and (3.55), the 6N bounding constraints (3.56)-(3.58), the $\frac{1}{2}M_{max}(M_{max} - 1)$ validity constraints (3.59), and the $M_{max}(N + 1) + N(3N + 1)$ integrality constraints in (3.60)-(3.64).

3.4.1.2 Result Achieved

A scalability analysis is performed testing both methods (i.e., for the matheuristic algorithm, indicated as $\mathcal{A}$, and for the exact method, indicated as $\widehat{\mathcal{A}}$) over instances with an increasing number of identical items. The items' dimensions are θ=300 mm (width), λ=200 mm (length), ψ=600 mm (height) and ω=10.8 kg (weight).

In the tests the pallet dimensions are equal to the standard EUR1 Euro pallet ones, i.e., Θ= 800 mm (width of the pallet), Λ=1200 mm (length of the pallet). The maximum admissible height for each bin is Ψ= 1800 mm, while the maximum weight supported by the pallet is F=1200 kg. These assumptions are realistic, especially for logistic companies that handle high quantities of goods.

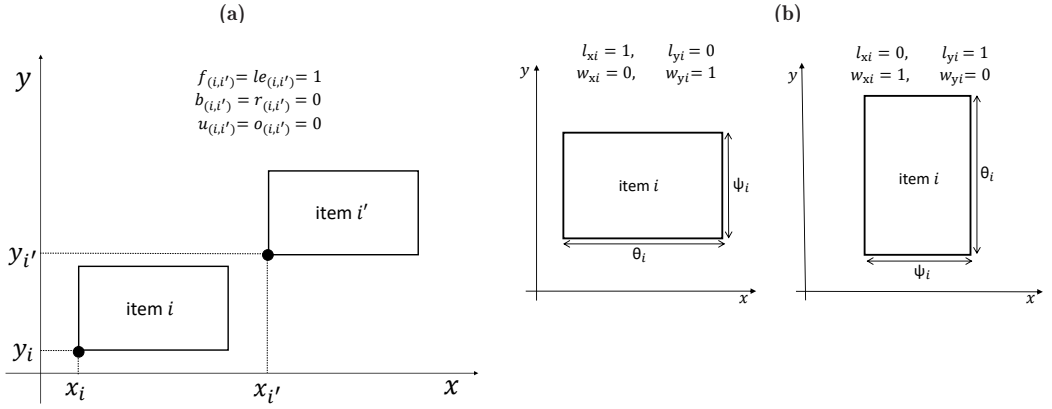


Figure 3.2: (a) Example of the relative position of two items laying in the x/y plane. (b) Horizontal placement of an item in the rotated and not rotated configuration.

Starting with an instance including 10 items and increasing at each test the number of items by one unit, the computed results reveal that in 10% of scenarios the exact solution provides a higher number of bins with respect to the matheuristics. On the contrary, the matheuristics succeeds in including all items in a single bin. Moreover, as shown in Fig. 3.3, the computational time of the matheuristics presents a growth rate significantly lower with respect to the exact solution. In particular, with 32 items the computational time of the matheuristics is 699.0% lower than the exact solution, thus demonstrating the efficiency of the proposed algorithm.

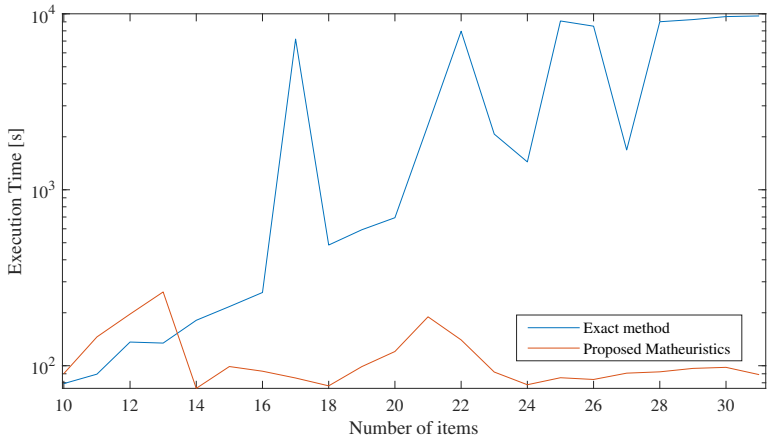


Figure 3.3: Execution time of the proposed algorithm ($\mathcal{A}$) and the exact method ($\widehat{\mathcal{A}}$) as a function of the items number.

Further, the performance of the proposed matheuristics are tested and compared with the ones of the exact method considering two additional scenarios, i.e., ScA and ScB. The two scenarios respectively include two and three types of different items, whose dimensions and weights are reported in Table 3.3. It is important to mention that in both scenarios the considered items can fit in a single bin. The obtained results, reported in Table 3.3 are evaluated in terms of computational time T_{ex}, average fill ratio $AvgFR_M$, and number of obtained bins **M** for the

matheuristic algorithm $\mathcal{A}$, and for the exact method $\widehat{\mathcal{A}}$. The outcomes show that in both cases the matheuristics can provide a solution with AvgFR_M=100% in shorter computational times with respect to the exact method, and both assign all items to a single bin.

Table 3.3: Set-up and results for the comparison of the proposed algorithm ($\mathcal{A}$) with the exact method ($\widehat{\mathcal{A}}$)

	Input						Output					
Scenario	**IDs**	**N**	λ_i **[mm]**	θ_i **[mm]**	ψ_i **[mm]**	ω_i **[kg]**	$\mathbf{T_{ex}}$		$\mathbf{AvgFR_M}$		**M**	
							$\mathcal{A}$	$\widehat{\mathcal{A}}$	$\mathcal{A}$	$\widehat{\mathcal{A}}$	$\mathcal{A}$	$\widehat{\mathcal{A}}$
ScA	2	18	400	{400,600}	{300,900}	{5.6,10.8}	2.6	851.84	100	100	1	1
ScB	3	26	{200,400}	{400,600}	{300,900}	{5.6,10.8}	5.4	1250.80	100	100	1	1

3.5 Comparison with a Literature Reference Matheuristics

In this subsection, the results achieved of the proposed matheuristics are compared with the ones obtained by in [11] using a set of randomly generated industrial instances. In the related literature, the principal well-known data-sets used to test the efficiency of 3D-SBSBPP algorithms lack most of the logistic data considered in this work (i.e., items' weight, stability index, supported load, overhang, categories, IDs, height and area gap, layer picking) that are necessary to practically implement the problem resolution. On the contrary, the work in [11] presents a matheuristics using a layer based column generation approach combined with second order cone programming and graphs, and includes all the requirements considered in this work except for the robotized layer picking. Moreover, this approach, considered as literature benchmark, is the extension of the one proposed by the same authors in [16], which proves that the algorithm largely outperforms the best performing literature's algorithms identified by [9], such as the ones in [22] and [23].

The data-set used for the comparison is obtained with the instance generator of [11] and is composed of 4 classes with 7 different instances including 100, 150, 200, 500, 1000, 1500, and 2000 items. The 4 classes contain different percentages of small to large volume items as specified in [16]. For what concerns the setup parameters, the pallet dimensions are Θ=1240 mm, Λ=840 mm and Ψ= 2200 mm, and F=1500 kg. Table 3.4 reports the average results achieved by testing each combination of class and number of items over five instances; the symbols $\mathcal{A}$ and $\mathcal{A}*$ are used to respectively indicate the proposed algorithm and the literature one. Columns I and II report the class of the instance (i.e., Classes 1 to 4) and the corresponding number of items (N); column III reports the minimum, the maximum, and the average number of bins (M) obtained with the two algorithms for the five instances; column IV reports the average CPU times obtained over the instances (AvgT_ex), and column V reports the minimum the maximum and the average value of the average bins' fill ratio (AvgFR_M) achieved over five instances. The achieved results show that this algorithm provides a higher fill ratio in terms of the average and the maximum values, while, in some cases, it provides a lower value of the minimum value with respect to the reference algorithm. In general, it is possible to notice that, on the one hand, the difference between the minimum and the maximum fill ratio values is higher for the proposed algorithm, meaning that it provides some very full configurations, and some other emptier because they are filled with the few remaining items. On the other hand, the number of filled bins is similar in the two methods, while the execution time is almost 70% higher with the proposed method in the small scale instances, while it is notably 170% lower in the large scale instances. Concluding, the developed comparison demonstrates that the proposed matheuristics can efficiently provide feasible bins' configurations with different set of instances both in terms of number of items and features heterogeneity. Moreover, the proposed method generally outperforms the one in [11] both

Table 3.4: Results for the comparison of the proposed algorithm ($\mathcal{A}$) with the literature benchmark [11] ($\mathcal{A}*$)

	N	M						AvgT_{ex} [s]		AvgFR_{M} [%]					
		min		max		avg				min		max		avg	
		$\mathcal{A}$	$\mathcal{A}*$	$\mathcal{A}$	$\mathcal{A}*$	$\mathcal{A}$	$\mathcal{A}*$	$\mathcal{A}$	$\mathcal{A}*$	$\mathcal{A}$	$\mathcal{A}*$	$\mathcal{A}$	$\mathcal{A}*$	$\mathcal{A}$	$\mathcal{A}*$
Class 1	100	1	1	2	2	1.4	1.8	21.90	6.02	78.2	28.47	89.23	59.40	80.5	35.37
	150	2	2	2	2	2	2	20.50	21.83	28.78	41.92	80.43	45.60	58.38	44.27
	200	3	3	5	3	3.3	3	3.50	40.89	16.51	37.88	83.44	39.71	53.69	39.07
	500	5	5	6	6	5.2	5.4	204.10	545.08	15.35	49.34	82.03	59.80	48.83	55.62
	1000	9	9	9	10	9	9.4	399.51	871.07	11.02	59.68	85.60	66.71	69.62	64.09
	1500	13	13	14	14	13.5	13.8	1254.19	2456.04	27,87	63.81	94,16	68.55	69,12	64.89
	2000	18	18	19	19	18.7	18.6	1744.52	3908.56	40.8	63.01	88.71	65.94	70.12	64.28
Class 2	100	2	2	2	2	2	2	21.02	3.27	26.4	29.05	85.4	30.46	75.4	29.74
	150	2	2	3	2	2.3	2	32.09	22.93	25.62	41.59	86.21	45.93	76.23	43.83
	200	2	2	3	3	2.5	2.8	29.13	47.25	47.32	38.16	81.32	59.93	68.46	43.29
	500	5	5	7	6	6.5	6.1	103.98	392.30	34.96	49.87	89.56	60.04	58.02	57.24
	1000	9	9	13	10	10.3	9.4	279.13	1080.03	14.56	58.9	82.33	65.56	57,84	62.97
	1500	14	13	14	14	14	13.4	1384.21	2158.65	37.31	63.81	85.23	68.55	70.45	66.68
	2000	18	18	19	19	18.6	18.4	1964.62	5201.74	20.54	62.87	80.32	66.46	68.58	64.76
Class 3	100	2	2	3	2	2.3	2	30.65	11.01	26.4	22.57	84.5	24.62	76.43	23.62
	150	2	2	3	2	2.3	2	61.32	10.45	43.25	34.4	87.43	36.53	63.36	35.57
	200	2	2	3	3	2.8	2.6	63.32	37.51	24.32	38.16	88.34	59.93	61.32	43.29
	500	5	4	7	5	6.1	4.2	206.41	392.30	20.52	47.27	80.79	60.67	66.68	57.16
	1000	8	8	9	8	8.4	8	212.23	1080.03	16.34	57.99	98,23	60.97	54.32	59.72
	1500	11	11	12	12	11.6	11.2	517.25	2661.04	44,54	59.58	88.23	65.84	52.77	63.9
	2000	15	15	16	16	15.7	15.8	1893.23	5201.74	19.00	59.48	80.91	63.42	53.63	60.62
Class 4	100	1	1	2	2	1.8	1.8	22.41	7.90	43.2	21.6	92.3	40.23	79.51	25.61
	150	2	2	3	2	2.5	2	36.01	17.07	49.9	31.07	87.2	34.18	64.4	32.47
	200	2	2	3	2	2.7	2	44.09	47.86	26.1	42.74	85.34	46.89	75.3	44.63
	500	4	4	5	5	4.6	4.4	114.33	451.83	15.98	44.42	83.63	56.01	57.06	51.36
	1000	8	8	9	10	9.1	8	577.21	1867.12	15.98	54.61	83.63	56.16	62.68	55.32
	1500	10	10	11	11	10.5	10.6	875.34	2158.65	37.31	60.34	85.23	67.31	70.45	63.35
	2000	15	15	16	16	15.6	15.4	2034.76	5201.74	20.49	55.23	84.91	59.08	79.63	57.4

in terms of computational time and fill ratio in large size industrial scenarios and consequently also the principal literature algorithms.

3.5.1 Tests on Industrial Data

In this subsection, the proposed matheuristic method is further tested on more complex scenarios based on real data provided by the Italian logistic company E80 Group [20]. These data are related to the three scenarios Sc1, Sc2, and Sc3 –described in Table 3.5– corresponding to logistic shipments with different level of item heterogeneity. The set-up of all scenarios is reported in Table 3.5, where column I specifies the scenario, column II the number of corresponding IDs, columns III-VI the interval for the length, width, height, and weight of the items included in the scenario. Each scenario is tested on 3 different instances of size 84, 486, and 522, corresponding to the three most common sets of deliveries. For the tests the pallet dimensions are assumed equal to the standard EUR1 Euro pallet ones, i.e., Θ= 800 mm (width of the pallet), Λ=1200 mm (length of the pallet). The maximum admissible height for each bin is Ψ= 1800 mm, while the maximum weight supported by the pallet is F=1200 kg.

Table 3.6 shows the performance evaluation of the matheuristics for each scenario: columns I and II report the scenario (i.e., Sc1, Sc2, Sc3) and the corresponding number of items (N); columns III and IV report the obtained number of layers (V) and bins (M); column V reports the total computational time (T_{ex}); column VI reports the average computational time needed by the algorithm to compute each bin (AvgT_{M}); column VII reports the average value of the average bins' fill ratio (AvgFR_{M}). It is apparent that the higher the item heterogeneity, the higher the

Table 3.5: Set-up for the test on industrial data

Scenario	**IDs**	λ_i **[mm]**	θ_i **[mm]**	ψ_i **[mm]**	ω_i **[kg]**
Sc1	80	[240,400]	[170,300]	[150,305]	[5.6,17]
Sc2	30	[240,400]	[170,300]	[150,305]	[5.6,17]
Sc3	11	[240,400]	[170,300]	[150,305]	[5.6,17]

computational time. In particular, the worst performance in terms of computational time is obtained in Sc1 that contains the lowest level of homogeneous items. Actually, in Sc1 the largest part of the execution time is spent in the Layer Building Phase of the matheuristics, since a high percentage of mixed layers is required to be computed.

Table 3.6: Results for the test on industrial data

Scenario	**N**	**V**	**M**	$\mathbf{T_{ex}}$**[s]**	$\mathbf{AvgFR_V}$ **[%]**	$\mathbf{AvgFR_M}$ **[%]**
Sc1	84	9	3	52	83.2	85.1
	486	69	13	306	84.1	54.3
	522	70	18	584.3	86.7	68.8
Sc2	84	10	2	91.1	88.8	63.7
	486	57	11	50.4	94.5	82.6
	522	62	16	66	99	99
Sc3	84	11	3	62.7	85.1	57.5
	486	59	16	26	97.5	89.3
	522	66	17	78.3	99	99

As for the quality of the solutions, first, the geometrical features of the composed bins are evaluated in terms of fill ratio. From Table 3.6 it can be noticed that the average fill ratio of the medium and large instances (i.e., with 486 and 522 items) is generally higher than the smallest one (i.e., with 84 items); in particular, for scenarios Sc2 and Sc3 it is up to the 99% of the total volume of the volume. In addition, it can be also noted that the average layers' fill ratio (AvgFR_V) is always higher than the average bins' fill ratio (AvgFR_M). Both results are due to two different reasons. First, the higher the number of items and the lower the number of IDs, the higher the number of created monoitem and monocategory layers (and consequently bins). Since these layers include items with the same geometric features and weight, the maximization of the fill ratio, i.e., eq. (3.1), can be more easily achieved with respect to the case of mixed items. Moreover, the algorithm is set so that the minimum fill ratio for the monoitem and monocategory layers must be higher than 90% and, in case this condition is not satisfied, the obtained layers are rejected and their items are used to create mixed layers. On the other hand, even if the average layers' fill ratio is higher due to weight, height, and safety constraints of the bins' building problem (Section 3.2.2), there can be limits to the possible configurations admissible for the bins' composition, especially in the case of mixed bins, thus the average bins' fill ratio can be lower than the average layers' fill ratio, thus leading to lower values in the average bins' fill ratio. To further highlight the difference in bins' fill ratio in the analyzed scenarios, Fig. 3.4 reports the 3D configuration of four illustrative examples that respectively represent: (a) a bin with residual mixed layers, (b) a full bin with mixed layers, (c) a full bin with heterogeneous monoitems layers, and (d) a full bin with homogeneous monoitem layers. The more homogeneous the items inside of the bin, the higher the fill ratio: the monoitem layers are generally the fullest ones (i.e., FR_mono is equal to 99% as specified in the initial set-up of the algorithm). Additionally, Fig. 3.5 reports examples respectively of mixed layers –i.e., (a) and (b)– and monoitem layers –i.e., (c)– configurations.

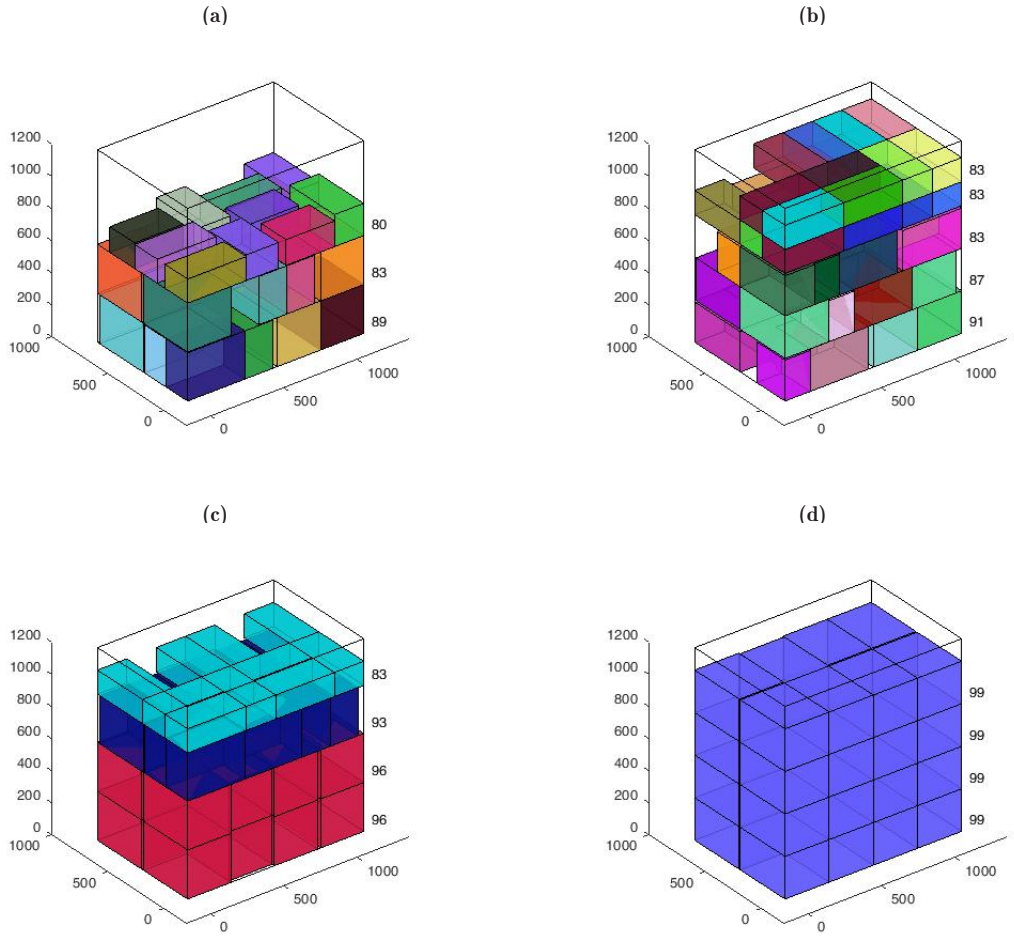


Figure 3.4: Examples of mixed bin with few items (a), mixed bin with heterogeneous items (b), mixed bin with homogeneous items (c), monoitem bin (d).

Note that the number reported inside each item represents the item ID.

Finally, the obtained results are assessed in terms of logistic requirements. In particular, the stability indices (i.e., $S1_k$ and $S2_k$) and the overhang index (i.e., $S3_k$) are evaluated for each layer of the composed bins. Table 3.7 shows AvgS1, AvgS2 and AvgS3 respectively representing the average value of $S1_k$, $S2_k$, and $S3_k$ over all the composed bins in the considered scenarios. According to [21] and [12], the higher the value of AvgS1 the higher the stability, while the lower the value of AvgS2 the higher the stability. For scenarios Sc2 and Sc3, very low values of AvgS2 are obtained, against a higher value in Sc1 that includes more mixed layers than the other two scenarios. As for AvgS1, it can be noticed that the corresponding values are particularly low, especially in scenarios Sc2 and Sc3. As a matter of fact, both in scenario Sc2 and Sc3 the bins' configurations are composed mainly by monoitem and monocategory layers, which are identical layers. Consequently, each item lies only on a totally full layer, thus not compromising the stability of the overall configuration (which has a high fill ratio and hence is more compact).

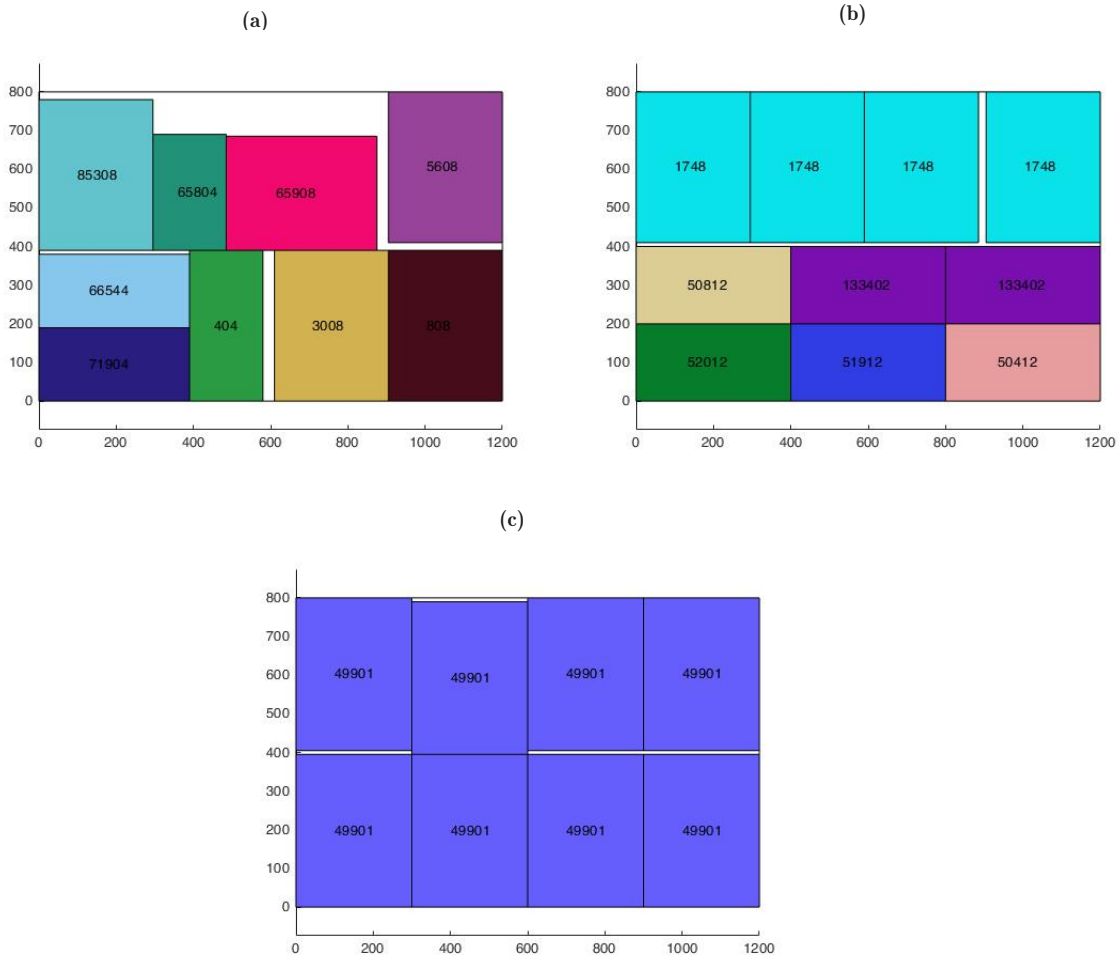


Figure 3.5: Examples of mixed layers (a, b) and a monoitem layer (c).

Lastly, the values obtained by AvgS3 demonstrate that the higher the number of IDs the higher the overhang index. This implies that the overhang feature remarkably contributes to improve the fill ratio of scenarios with item heterogeneity, thus providing benefit to the bin packing in realistic scenarios.

3.6 Conclusion

The 3D Bin Packing Problem (3D-BPP) has a crucial role in Industry 4.0 and in particular in the management of internal logistics, since it allows to save time and resources in the mobilization of goods. Consequently, the 3D-BPP is largely studied in the literature both because of its NP-hard nature and its high versatility in industrial applications. This work presents an innovative matheuristic algorithm based on a layer-building approach that allows the automated resolution of the 3D-BPP in a short computational time and suitably for the industrial context. In particular, it has been proposed a mixed integer non-linear programming problem to formulate the 3D-BPP

Table 3.7: Stability and overhanging indices

Scenario	N	AvgS1	AvgS2 [%]	AvgS3
Sc1	84	1.41	21.08	0.47
	486	1.31	18.83	0.44
	522	1.23	17.21	0.38
Sc2	84	1.17	3.93	0.29
	486	1.13	3.24	0.23
	522	1.08	3.01	0.20
Sc3	84	1.05	2.50	0.18
	486	1.02	2.00	0.16
	522	1.01	1.90	0.13

including a complete set of industrial requirements and a matheuristic algorithm is presented to efficiently solve the problem. Simulation results on both realistic and real data prove the efficiency and effectiveness of the proposed algorithm in terms of computational time, optimization of the bins' configuration and number, and stability of the bins. Furthermore, the proposed algorithm outperforms the results of the respective exact method. Future developments will consider the extension of the proposed approach to the multiple bin size bin packing problem, i.e., the case of multiple types of load aids with different sizes, and the inclusion of further logistic constraints that can improve the stability of the packed bins and shape constraints. Moreover, with the aim of implementing an approach even more suitable for the industrial sector, the implementation of a multi-objective optimization approach to generate Pareto efficient solutions will be considered, followed by a multi-criteria analysis to rank these solutions.

References

[1] Rüßmann, M., Lorenz, M., Gerbert, P., *et al.*, "Industry 4.0: The future of productivity and growth in manufacturing industries," *Boston Consulting Group*, vol. 9, no. 1, pp. 54–89, 2015.

[2] Lasi, H., Fettke, P., Kemper, H.-G., Feld, T., and Hoffmann, M., "Industry 4.0," *Business & information systems engineering*, vol. 6, no. 4, pp. 239–242, 2014.

[3] Strandhagen, J. O., Vallandingham, L. R., Fragapane, G., Strandhagen, J. W., Stangeland, A. B. H., and Sharma, N., "Logistics 4.0 and emerging sustainable business models," *Advances in Manufacturing*, vol. 5, no. 4, pp. 359–369, 2017.

[4] Lee, C., Lv, Y., Ng, K., Ho, W., and Choy, K., "Design and application of internet of things-based warehouse management system for smart logistics," *International Journal of Production Research*, vol. 56, no. 8, pp. 2753–2768, 2018.

[5] Wang, F. and Lim, A., "Effective neighborhood operators for solving the flexible demand assignment problem," *IEEE Transactions on Automation Science and Engineering*, vol. 5, no. 2, pp. 289–297, 2008.

[6] Cavone, G., Carli, R., Troccoli, G., Tresca, G., and Dotoli, M., "A milp approach for the multi-drop container loading problem resolution in logistics 4.0," in *2021 29th Mediterranean Conference on Control and Automation (MED)*, 2021, pp. 687–692. DOI: 10.1109/MED51440.2021.9480359.

[7] Cavone, G., Dotoli, M., and Seatzu, C., "Management of intermodal freight terminals by first-order hybrid petri nets," *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pp. 2–9, 2015.

[8] Cavone, G., Dotoli, M., and Seatzu, C., "Resource planning of intermodal terminals using timed petri nets," in *2016 13th International Workshop on Discrete Event Systems (WODES)*, IEEE, 2016, pp. 44–50.

[9] Zhao, X., Bennell, J. A., Bektaş, T., and Dowsland, K., "A comparative review of 3d container loading algorithms," *International Transactions in Operational Research*, vol. 23, no. 1-2, pp. 287–320, 2016.

[10] Fischetti, M. and Fischetti, M., "Matheuristics," in *Handbook of heuristics*, Springer, 2018, pp. 121–153.

[11] Gzara, F., Elhedhli, S., and Yildiz, B. C., "The pallet loading problem: Three-dimensional bin packing with practical constraints," *European Journal of Operational Research*, vol. 287, no. 3, pp. 1062–1074, 2020.

[12] Moura, A. and Oliveira, J. F., "A grasp approach to the container-loading problem," *IEEE Intelligent Systems*, vol. 20, no. 4, pp. 50–57, 2005.

[13] Hifi, M., Kacem, I., Nègre, S., and Wu, L., "A linear programming approach for the three-dimensional bin-packing problem," *Electronic Notes in Discrete Mathematics*, vol. 36, pp. 993–1000, 2010.

[14] Paquay, C., Limbourg, S., Schyns, M., and Oliveira, J. F., "Mip-based constructive heuristics for the three-dimensional bin packing problem with transportation constraints," *International Journal of Production Research*, vol. 56, no. 4, pp. 1581–1592, 2018.

[15] Zhao, H., She, Q., Zhu, C., Yang, Y., and Xu, K., "Online 3d bin packing with constrained deep reinforcement learning," *arXiv preprint arXiv:2006.14978*, 2020.

[16] Elhedhli, S., Gzara, F., and Yildiz, B., "Three-dimensional bin packing and mixed-case palletization," *Informs Journal on Optimization*, vol. 1, no. 4, pp. 323–352, 2019.

[17] Chen, C. S., Lee, S. M., and Shen, Q. S., "An analytical model for the container loading problem," *European Journal of Operational Research*, vol. 80, no. 1, pp. 68–76, 1995.

[18] Wikipedia, *C sharp (programming language)*. [Online]. Available: https://en.wikipedia.org/wiki/C_Sharp_(programming_language).

[19] OR-Tools, G., *The glop linear solver*. [Online]. Available: https://developers.google.com/optimization/lp/glop.

[20] *E80 group*, Available on line, 2024. [Online]. Available: https://www.e80group.com/it/.

[21] Bischoff, E. E. and Ratcliff, M., "Issues in the development of approaches to container loading," *Omega*, vol. 23, no. 4, pp. 377–390, 1995.

[22] Martello, S., Pisinger, D., and Vigo, D., "The three-dimensional bin packing problem," *Operations research*, vol. 48, no. 2, pp. 256–267, 2000.

[23] Crainic, T. G., Perboli, G., and Tadei, R., "Ts2pack: A two-level tabu search for the three-dimensional bin packing problem," *European Journal of Operational Research*, vol. 195, no. 3, pp. 744–760, 2009.

Chapter 4

A Matheuristic Algorithm for the Configuration of Automated Vertical Lift Modules Warehouses

Abstract

Nowadays, thanks to the advent of digitalization, logistic companies have experienced a rapid increase in product demand and the expansion of production scales. This implies the need for innovative and more efficient stocking solutions. In this chapter, the focus is on Vertical Lift Module (VLM) warehouses, which are a technologically advanced automated solution still not fully exploited in this context. The VLMs are based on the use of sliding trays to implement the so-called parts-to-pickers solution and are particularly useful for the storage of reduced dimensions items. Moreover, a VLM may require up to 90% less surface area than a corresponding traditional warehouse. Thanks to their versatility and compactness, they can find potential usage not only in industrial environments (e.g., storage of working tools) but also for retail. For logistic companies, the design of a VLM is a non-trivial activity, which is often manually developed in a trial-and-error fashion. In particular, the dimensions, internal partitioning, and allocation of each tray in the VLM must be properly selected to avoid space loss while satisfying various logistic constraints. In this context, the contribution of the chapter consists in the definition of three Mixed-Integer Linear Programming models, which are here defined to address the trays' internal configuration and the trays' allocation problems, and a two-phase matheuristic algorithm (i.e., an algorithm that combines the use of exact mathematical methods and heuristics) to streamline and automate the design of the VLM internal configuration. The proposed algorithm receives as input the features of a set of items to be allocated inside the VLM, a predetermined set of trays' types (i.e., characterized by different geometric dimensions), parameters necessary to set the constraints of the problem, and a priority rule for the allocation of trays, while it provides as output the most space-efficient configuration of the VLM. In particular, the algorithm provides the logistic operator with (1) the necessary types and quantities of trays, (2) their internal partitioning, (3) the proper position of the items in each tray, and (4) the position of each tray in each shelving of the VLM. In the chapter, an extensive test campaign is performed and demonstrates the effectiveness of the algorithm under various realistic operative scenarios. Moreover, a wide set of priority rules is presented and compared, to support the logistics operator in selecting the most performing one depending on the considered scenario.

Contents

4.1 Introduction

The last decades have seen the establishment of Logistic 4.0 foundations, which are leading to the improvement of traditional logistic methods by the integration of innovations brought by systems digitization and automation [1], [2]. In this context, automated warehouses take place due to the several advantages that can bring to the internal logistics sector such as a high rate of warehouse space utilization, automated management, labor savings, and fast and accurate operations of inputs and outputs. The application of automation to warehouses can improve all of their main activities, i.e., (I) reception and management of products and customer orders, (II) storage of products, (III) shipment of products, and (IV) picking of orders. Particular attention is devoted to this last activity, i.e., the process of picking products from the warehouse based on customer demand, due to its great influence on distribution centers' performance. According to Dallari et al. [3], a dedicated Order Picking System (OPS) can be conceived and implemented for the optimization of this activity. The authors classify OPSs mainly into static and dynamic solutions. The first type is characterized by the storage of goods in racks or devices that have a fixed location, so these are usually simple to implement and inexpensive. The latter type instead is characterized by the movement of goods to the picker (i.e., the operator) and automated systems and computer software tools are used for supporting the correct functioning of the storage system, thus their design and operation are more complex.

In this chapter, the focus is on dynamic OPS solutions and in particular, on Vertical Lift Modules (VLMs) warehouses that on the one hand improve the labor conditions thanks to the implementation of the so-called "the goods to the man" principle, and on the other hand, can ensure greater compactness with respect to traditional warehouses and great versatility, e.g., they can be used for tool storage, spare parts storage, shipment warehouse, and retail market. In general, VLMs warehouses are closed structures that contain several shelvings organized into columns and holding a set of sliding trays (see the illustrative immage reported in Fig.4.1). The trays are partitioned into sectors to hold items and are moved by a liftmounted module that travels between the shelvings in order to make the specific tray available to the operator. This device is driven by an automatic control system, which interfaces with dedicated software, in order to set the correct order for picking trays [4], [5]. For logistic companies, the design of a VLM is a non-trivial and time-consuming activity, which is often manually developed, i.e., without the support of ad-hoc software and tools, and requires experienced operators and various iterations. In particular, the proper selection of the dimensions, internal partitioning, and allocation of each tray in the VLM, to avoid space loss while satisfying various logistic and operational constraints, is often developed in an unstructured and naive way.

The objective of this work is to define a matheuristic algorithm (i.e., a combination of exact resolution methods and the use of heuristics), for the efficient and automated design of the configuration of the VLM. The algorithm must be able to: (I) define the most efficient allocation of the items in the trays, that minimizes the unused space in the warehouse while satisfying various design constraints, such as weight limits, rotation of the items, and the definition of a limited number of trays internal configurations with the assignment of separators to subdivide the items; (II) allocate the full trays in the minimum number of columns taking into account different priority rules defined by the company, and eventually allocate empty trays to complete the columns configurations. The remainder of this chapter is organized as follows:

- Section 4.2 describes the problem statement, i.e., the description of the problems assumptions and requirements in the design of VLMs warehouses;
- Section 4.3 presents a detailed explanation of the proposed two-phase matheuristic algorithm and its steps;

- Section 4.4 describes the proposed mathematical formulations for the trays internal configuration and allocation in the VLM problems, detailing the objectives to be achieved and the constraints to be met in accordance with the assumptions and requirements defined in Section 4.2;
- Section 4.5 reports the tests performed with the proposed algorithm, with the aim of validating the exact mathematical models and the entire algorithm;
- Section 4.6 provides the final considerations regarding the proposed solution and possible future research developments.

4.2 Problem Statement

In the design of a VLM several problems need to be tackled in order to match and personalize the request of each warehouse: the first of them is the selection of the proper location of the warehouse. Such a decision depends mainly on the destination of use and obviously influences the design and related requirements. For example, if the VLM has to be placed outdoors, its dimensions are more flexible as they depend on the available space in the installation site, while if it has to be placed indoors its dimensions are less flexible, as the dimensions of the columns are constrained by the features of the building where the VLM has to be located. After selecting the location of the warehouse to be designed, its final dimensions depend on: (1) the volume and the total number of items that the customer needs to store; (2) the number of replenishment/picking bays, also called access bays, where the operator can access the trays, which reduce the available space for placing trays and items.

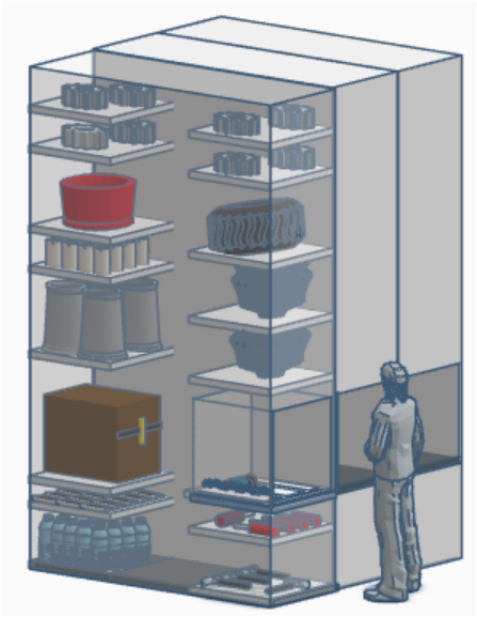


Figure 4.1: Illustrative image of a VLM warehouse.

The attention in this thesis is focused on the design of VLMs that have to be installed indoors and, in order to create the most accurate and complete algorithm possible, expert producers of logistics automation and VLM systems were surveyed to properly define the necessary logistic requirements.

It is assumed that the trays must have specific width and depth predefined by the company and depending on the features of the VLM. The choice of the size of trays depends mainly on the size of the items that must be allocated and therefore also on the possibility to place items in an optional or mandatory direction. The internal space of the trays can be separated through the positioning of dividers, both along the longitudinal and transversal axes of the tray, the space delimited by dividers is defined as a sector. The positioning of the dividers along both axes can not be done arbitrarily, but they have to be placed at predetermined positions multiple of two different pitches, one for each axis. Moreover, the transversal dividers must be of the same length as the tray, while the longitudinal dividers can start from one transversal divider and reach one of the borders (see e.g., Figure 4.2, where longitudinal dividers are reported in red color, while transversal dividers are reported in blue color). The different combinations in which the transversal and longitudinal dividers are placed inside each tray constitute a “pattern", and even if the combinations can be manifold, it is required that only a limited number of patterns can be defined. Figure 4.2 represents one possible pattern, with two transversal dividers and one full-length longitudinal divider, and another longitudinal one that extends only partially on the tray. Note that, as reported in the figure, in the remainder of the chapter the x and y axes respectively refer to the length and the width of the tray (and thus the VLM), while the height corresponds to the z-axis.

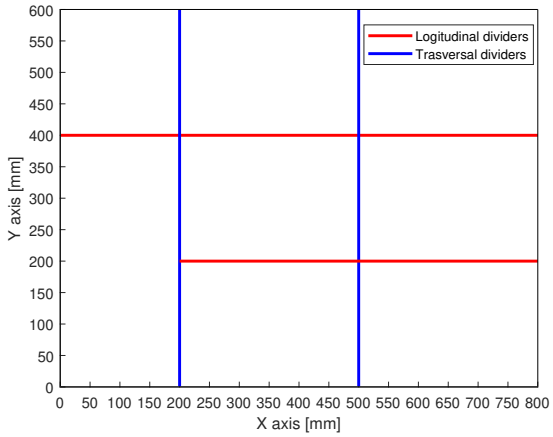


Figure 4.2: Example of pattern in a top-view bi-dimensional representation of the tray.

The business rules taken into account in the definition of the proposed matheuristics are as follows:

1. minimization of unused volume inside each tray;
2. minimization of the number of trays used;
3. minimization of unused column space by inserting also empty trays (thus giving the possibility to expand the warehouse after the design of the VLM);
4. the heights of the tray can be chosen among a finite number of available ones;
5. items can be rotated along the three axes;
6. the sum of the weight of items, tray, and equipment (dividers and borders) must be lower than the maximum sustainable weight of the shelving;

7. the dividers have to be placed at their predetermined widths and length, according to the pitch defined by the company's industrial requirement. Moreover, the transversal dividers must be of the same length as the tray, while the longitudinal ones can start from one of the transversal dividers and end at the border of the tray;
8. the volume occupied by the dividers and the gripping space (i.e., the space required for picking up an object from its position within a tray) must be taken into account;
9. items of the same kind must be assigned to sectors thereof size;
10. between two consecutive trays a fixed vertical minimum clearance must be considered;
11. the number of patters of trays (i.e., allocation of dividers in the tray) is limited;
12. there must be space between trays at least equal to the space required for the pick-up mechanism;
13. the number of trays per type, i.e, the number of trays with the same height, in each column must respect the limitation given by the priority criterion defined by the selected priority rule
14. the frequency of empty trays must be lower than the frequency of full trays, for each type of tray.

It is highlighted that a priority rule defines the order in which the trays should be placed inside the VLMs columns, and its choice can be done among a wide set of alternatives (details on possible alternatives are provided in Section 4.5), so as to provide flexibility in the application in different operational scenarios. An example of a priority rule can be the placement of trays in accordance with a height-decreasing order.

4.3 The Proposed Matheuristics for VLMs Configuration

In this section, first, a high-level description of the different steps of the proposed two-phase matheuristic algorithm is presented. Then, a detailed description of each phase and the related sub-phases is provided. It is worth to highlight that the related optimization problems are detailed in the subsequent Section 4.4.

As shown in Figure 4.3, the algorithm is characterized by the following 2 phases that have to be executed sequentially:

1. **Trays' internal configuration phase:** devoted to the definition of the proper configuration of the items inside the trays according to the business rules. It takes as input the list of the items and the VLM's configuration parameters and computes as output the list of filled trays and their configuration. It is composed of two different sequential sub-phases:
   - *Mono-item trays sub-phase*
   - *Mixed-items trays sub-phase*
2. **Trays' allocation phase:** devoted to the composition of the VLM's columns according to the priority rules. It takes as input the filled trays computed with the trays' internal configuration phase and provides as output their configuration into one or more columns. This phase is composed of three sub-phases:
   - *Pre-processing sub-phase*
   - *Processing sub-phase*
   - *Post-processing sub-phase*

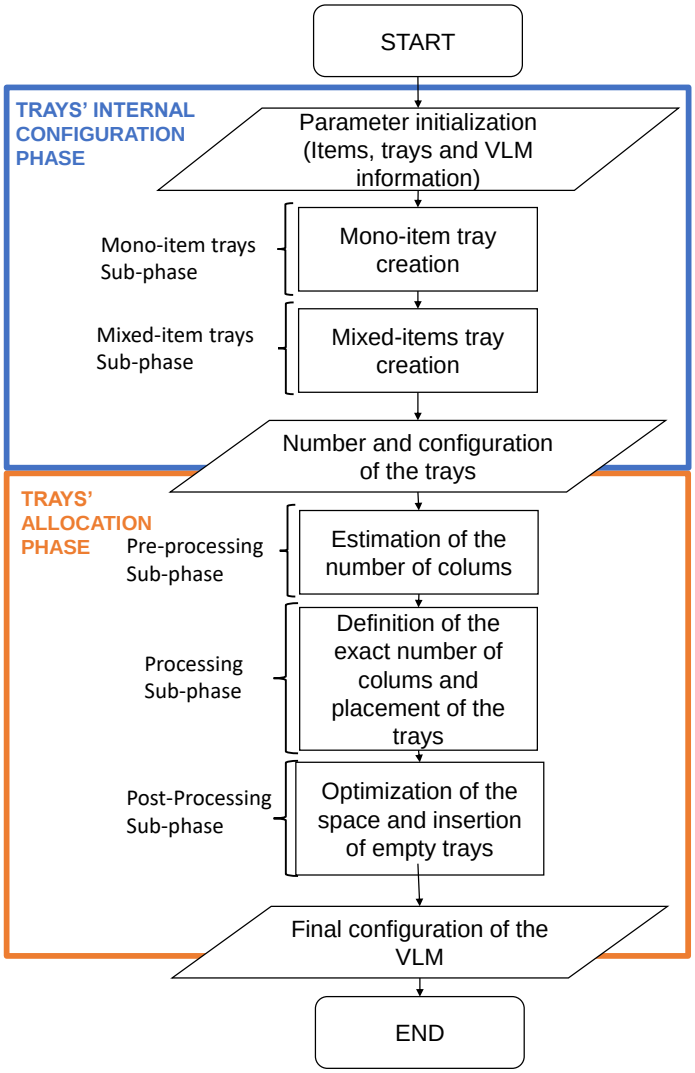


Figure 4.3: The proposed two-phase matheuristics algorithm.

4.3.1 Trays' Internal Configuration Phase

This is the first phase of the algorithm and it is devoted to the definition of the positioning of the items inside the tray. To this aim, a selection of the proper trays' heights among the available ones has to be performed. Then, the patterns of separators inside the trays have to be defined, together with the positions of the items. In order to speed up the algorithm, this phase is decomposed into two different sub-phases: the mono-item trays sub-phase that executes the initialization of data and reduces the computational complexity and the number of items to be processed by the subsequent sub-phase, by creating trays with just one type of item (i.e., mono-item trays), and the mixed-items trays sub-phase that computes the configuration of the mixed-items trays. The final outcome of this phase is the definition of the number of full trays and their internal configuration. Details are as follows.

A.1. Mono-item Trays Sub-phase.

First, the input parameters are set, i.e., the number N of items, the length l_i, width w_i, height h_i, and weight w_i of each i-th item, the width W length L and weight C that are the same for all the j-th trays, the admissible heights H_k with $k \in$K of the trays, the maximum weight $C_{\max}$ supported by the tray, the pitches dimensions s_x and s_y, the thickness t and the volume densities ρ_x and ρ_y of the dividers and edges along the two axes, the grip space g_i for each i-th item, the admissible number of patterns P. Then the algorithm verifies if there are some input items whose dimensions do not fit in the VLM: if they exceed the highest trays' available height, or have some dimensions that do not fit the machine parameters they are discarded, otherwise, if they exceed the maximum tray splitting width they are considered for the definition of a tray containing a single item. After this check, the three dimensions of the items are considered again analyzing the items of the same type singularly. The algorithm calculates N_t, as the integer approximate number of items contained in a single tray considering the area of the widest of the six faces of the item and the area of the tray. This assumption allows for avoiding the assignment of the highest dimension of the items to the height of the trays and consequently allows the insertion in the VLM columns of a higher number of trays, moreover, it allows for reducing the computational complexity of the optimization problem of subsection 4.3.1 by providing in advance the values for some of the binary variables related to rotation and thus reducing the solution space. Then, with N_t and the total number of items of the same type, the sub-phase checks if there are enough items to create at least one mono-item tray (and thus the correlated pattern), or otherwise if the items have to be inserted in the list of items to be allocated in mixed-items trays. Note that in the list are also included the exceeding items that are not assigned to mono-item trays, when the corresponding set presents a number of items that is not a proper multiple of N_t. Both the configuration of mono-item and mixed-items trays is obtained by solving the MINLP problem of subsection 4.3.1. The output of this sub-phase are: the assignment of an item i to a specific tray j indicated by p^*_{ij}, the coordinates of the left bottom corner of each item in the corresponding trays indicated by (x^*_i, y^*_i, z^*_i), the height of the j-th tray indicated by $\eta^*_{j,k}$, the orientation of each i-th item, indicated by l^*_{xi} i.e., the length of the item is parallel to the x axis, w^*_{yi}, i.e., the width of the item is parallel to the y axis, the number of defined patterns P_1, and the list containing all the non-allocated items and the corresponding parameters.

A.2. Mixed-items Trays Sub-phase.

This sub-phase, starting with the list of residual items (i.e., all the items which are not assigned to any tray) obtained from the previous sub-phase and the number of patterns P_1, performs the clustering of the items by applying the k-means algorithm on two parameters: the length and width of each item. The number of clusters is defined as the difference between the number of patterns required by the company (P) and the number of patterns obtained in the previous sub-phase (P_1). Next, similarly to the previous sub-phase, the algorithm calculates N_c (i.e., the number of items of the same cluster contained in a single tray) considering the dimensions of the biggest item. If N_c is sufficient to create a single tray, the algorithm solves the MINLP model of Section 4.4, otherwise, if the number of items is not sufficient or the configuration obtained is not satisfactory, the items are collected in the list including all the remaining items. This operation is performed iteratively for each produced cluster until all the items are assigned to a mixed-items tray. It is worth noticing that N_c is calculated considering the biggest item of the cluster because allows taking into account the space loss due to separators and the gripping requirements. After this, the trays' internal configuration phase is concluded and the information

regarding the obtained trays, i.e., the number of trays for each available height (Z_k, with k=1,.., K), and the corresponding configuration of the items and separators are given as output i.e., for each input item the values p^*_{ij}, x^*_i, y^*_i, z^*_i, $\eta^*_{j,k}$,l^*_{xi}, l^*_{zi}, w^*_{yi}, and h^*_{zi} are provided. Moreover, for the sake of practical applicability, a picture of the disposition of the items and the dividers inside each tray is provided to the operator.

4.3.2 Trays' Allocation Phase

This phase takes as input the number and dimensions of the trays (T, K, and Z_k) defined in the trays' internal configuration phase and executes the corresponding three sub-phases, i.e., pre-processing, processing, and post-processing sub-phases, so as to define the final configuration of the VLM and thus the ordering of the trays into each column, ensuring the highest possible occupation.

B.1. Pre-processing Sub-phase.

This sub-phase starts by sorting the input trays by height (H_k) in decreasing order. Then the approximate number of columns is calculated. In particular, a counter takes track of the number of filled columns: through a loop, the trays, ordered by height in descending order are inserted one at a time in the column, if the tray fits the empty space in the columns it is placed and removed from the list, otherwise, if the height of the tray to insert exceeds the maximum allowable height of a column, the counter of the number of columns is incremented and a new column is considered. This operation is repeated until the number of trays to insert is zero and thus the outcome of this pre-processing operation is an approximate maximum number of columns to fill, which will be used by the priority rules for selecting the right quantity of trays.

B.2. Processing Sub-phase.

Once obtained the number of columns in the pre-processing sub-phase, this sub-phase aims at determining the exact position of the trays inside each column. In particular, the allocation of the filled trays in each of the columns is performed according to the priority rule set at the beginning of the algorithm, which then influences the frequency ϕ_1, the threshold Φ, and the number of tray per type Z_r. Subsequently, for each column, the mathematical model for the allocation of the filled trays, which is detailed in the following Section 4.4.2.1, is executed to define the final allocation of the trays represented with the variable $z*_{1r}$.

B.3. Post-processing Sub-phase.

The objective of this sub-phase is to exploit the potentiality of the whole VLM's structure by reducing the empty space left after the placement of the filled trays. So, in this sub-phase, given the maximum number of columns calculated in the pre-processing sub-phase, the composition of the columns filled with the filled trays obtained in the processing sub-phase, the optimization of the total space is executed by choosing the correct type and then adding the empty trays into each column. This operation is performed iteratively using the mathematical model for the optimization of the empty trays, which is detailed in the following Section 4.4.2.2. The end of this phase coincides with the end of the whole algorithm, which provides as output the whole configuration of the VLM both in terms of the internal configuration of the trays and their disposition in the VLM's columns (i.e., the values of $z*_{1r}$ and $z*_{2r}$). It is worth mentioning that, in order to increase the usability of the algorithm the two phases can be executed singularly so that the company can choose to re-optimize only one of the two aspects.

4.4 Mathematical Formulations of the VLM's Configuration Problem

This section presents the models formulated for the proposed algorithm. In particular, the first subsection presents the formulation of the optimization problem regarding the internal configuration of the trays, while the second subsection presents the formulation of two optimization problems regarding the allocation of the trays into columns.

It is worth mentioning that the following formulations are defined in a general way, without considering the distinction between monoitem and mixed trays (for the first subsection) and the business rule used (for the second subsection) since both do not depend strictly on the type of input parameters.

4.4.1 Trays' Internal Configuration Problem

This subsection presents the proposed formulation for the trays' internal configuration problem. In particular, the model is first formulated as a Mixed Integer Non-Linear Programming (MINLP) model and then is linearized by means of the McCormick envelop convex relaxation, as reported in Subsection 4.4.1.1 Table 4.1 and Table 4.2 reports the list of parameters, while Table 4.3 reports the list of variables used in the problem formulation.

Table 4.1: Common Parameters

Parameter	Description
N	Number of items
T	Number of trays
H_k	Values of the admissible trays' heights

Table 4.2: Tray's internal configuartion Parameters

Parameter	Description
Trays' internal configuration parameters	
M	A big arbitrary number
K	Maximum number of different heights of a tray
l_i	Length of the i-th item
w_i	Width of the i-th item
h_i	Height of the i-th item
m_i	Weight of the i-th item
W	Width of the trays
L	Length of the trays
C	Weight of the trays
C_{max}	Maximum weight supported by a tray
t	Thickness of dividers and edges
s_w	Pitch of the dividers along the width of the tray
s_l	Pitch of the dividers along the length of the tray
ρ_x, ρ_y	Volume density of the dividers and edges
g_i	Grip space for i-th item
α_1,α_2	Weight of the first/second term of the objective function

The obtained problem may be written as follows:

Table 4.3: Trays' internal configuration variables

Variable	Description
i	Index of the i-th item
j	Index of the j-th tray
k	Index of the k-th possible height
p_{ij}	$\begin{cases}1 & \text{Item } i \text{ is inside the tray } j\\ 0 & \text{otherwise}\end{cases}$
η_{jk}	$\begin{cases}1 & \text{height } k \text{ assigned to tray } j\\ 0 & \text{otherwise}\end{cases}$
(x_i, y_i, z_i)	Coordinates of the bottom left front corner of item i
l_{xi}	$\begin{cases}1 & \text{Length of } i\text{-th item parallel to the } x \text{ axes}\\ 0 & \text{otherwise}\end{cases}$
l_{zi}	$\begin{cases}1 & \text{Length of } i\text{-th item parallel to the } z \text{ axes}\\ 0 & \text{otherwise}\end{cases}$
w_{yi}	$\begin{cases}1 & \text{Width of } i\text{-th item parallel to the } y \text{ axes}\\ 0 & \text{otherwise}\end{cases}$
$a_{ii'}$	$\begin{cases}1 & \text{Item } i \text{ is on the left of item } i'\\ 0 & \text{otherwise}\end{cases}$
$b_{ii'}$	$\begin{cases}1 & \text{Item } i \text{ is on the right of item } i'\\ 0 & \text{otherwise}\end{cases}$
$c_{ii'}$	$\begin{cases}1 & \text{Item } i \text{ is in front of item } i'\\ 0 & \text{otherwise}\end{cases}$
$d_{ii'}$	$\begin{cases}1 & \text{Item } i \text{ is behind item } i'\\ 0 & \text{otherwise}\end{cases}$
σ_i	$\begin{cases}1 & \text{The grip space is assigned along the } x \text{ axes}\\ 0 & \text{The grip space is assigned along the } y \text{ axes}\end{cases}$
f_i	Dimension of item i along the x-axis
D_i	Slack variable for the separation in sectors
q_{1i}, q_{2i}	Quantization of the range of x_i,y_i
$\begin{cases}\Delta^1_{ii'}, \Delta^2_{ii'}\\ \Gamma^1_{ij'}, \Gamma^2_{ij'}\end{cases}$	$\begin{cases}\text{Auxiliary variables for the linearization}\\ \text{of the non-linear cross products}\end{cases}$

$$\min[\alpha_1 \ \mathrm{WL}(\sum_{j=1}^{\mathrm{T}}\sum_{k=1}^{\mathrm{K}}\eta_{jk}\mathrm{H}_k) + \alpha_2(\sum_{j=1}^{\mathrm{T}}\sum_{k=1}^{\mathrm{K}}\eta_{jk})] \tag{4.1}$$

s.t.

$$x_i = \mathrm{s_w} q_{1i}, \forall i \in \{1, \ldots, \mathrm{N}\} \tag{4.2}$$

$$y_i = \mathrm{s_l} q_{2i}, \forall i \in \{1, \ldots, \mathrm{N}\} \tag{4.3}$$

$$\sum_{j=1}^{\mathrm{T}} p_{ij} = 1, \forall i \in \{1, \ldots, \mathrm{N}\} \tag{4.4}$$

$$\sum_{k=1}^{\mathrm{K}} \eta_{jk} \leq 1, \forall j \in \{1, \ldots, \mathrm{T}\} \tag{4.5}$$

$$f_i = \mathrm{l}_i l_{xi} + \mathrm{w}_i(l_{zi} - w_{yi} + h_{zi}) + \mathrm{g}_i\sigma_i + \mathrm{t} + \mathrm{h}_i(1 - l_{xi} - l_{zi} + w_{yi} - h_{zi}) \tag{4.6}$$

$$x_i + D_i \leq x_{i'} + (1 - a_{ii'})\mathrm{M}, \forall i, i' \in \{1, \ldots, \mathrm{N}\}, i \neq i' \tag{4.7}$$

$$x_{i'} + D_{i'} \leq x_i + (1 - b_{ii'})\mathrm{M}, \forall i, i' \in \{1, \ldots, \mathrm{N}\}, i \neq i' \tag{4.8}$$

$$D_i \geq f_i, \forall i \in \{1, \ldots, \mathrm{N}\} \tag{4.9}$$

$$D_i \geq f_{i'}(1 - a_{ii'} - b_{ii'}), \forall i, i' \in \{1, \ldots, \mathrm{N}\}, i \neq i' \tag{4.10}$$

$$y_i + \mathrm{l}_i(1 - l_{xi} - l_{zi}) + \mathrm{w}_i w_{yi} + \mathrm{h}_i(l_{xi} + l_{zi} - w_{yi}) + \mathrm{g}_i(1 - \sigma_i) + \mathrm{t} \leq y_{i'} + (1 - c_{ii'})\mathrm{M}, \forall i, i' \in \{1, \ldots, \mathrm{N}\}, i \neq i' \tag{4.11}$$

$$y_{i'} + \mathrm{l}_{i'}(1 - l_{xi'} - l_{zi'}) + \mathrm{w}_{i'} w_{yi'} + \mathrm{h}_{i'}(l_{xi'} + l_{zi'} - w_{yi'}) + \mathrm{g}_{i'}(1 - \sigma_{i'}) + \mathrm{t} \leq y_i + (1 - d_{ii'})\mathrm{M}, \forall i, i' \in \{1, \ldots, \mathrm{N}\}, i \neq i' \tag{4.12}$$

$$a_{ii'} + b_{ii'} + c_{ii'} + d_{ii'} \geq (p_{ij} + p_{i'j}) - 1, \forall i, i' \in \{1, \ldots, \mathrm{N}\}, \forall j \in \{1, \ldots, \mathrm{T}\}, i \neq i' \tag{4.13}$$

$$x_i + D_i \leq \mathrm{L}_k + (1 - p_{ij})\mathrm{M}, \forall i \in \{1, \ldots, \mathrm{N}\}, \forall j \in \{1, \ldots, \mathrm{T}\} \tag{4.14}$$

$$y_i + \mathrm{l}_i(1 - l_{xi} - l_{zi}) + \mathrm{w}_i w_{yi} + \mathrm{h}_i(l_{xi} + l_{zi} - w_{yi}) + \mathrm{g}_i(1 - \sigma_i) + \mathrm{t} \leq \mathrm{W} + (1 - p_{ij})\mathrm{M}, \forall i \in \{1, \ldots, \mathrm{N}\}, \forall j \in \{1, \ldots, \mathrm{T}\} \tag{4.15}$$

$$z_i + \mathrm{l}_i l_{zi} + \mathrm{w}_i(1 - l_{zi} - h_{zi}) + \mathrm{h}_i h_{zi} \leq \eta_{jk}\mathrm{H}_k + (1 - p_{ij})\mathrm{M}, \forall i \in \{1, \ldots, \mathrm{N}\}, \forall j \in \{1, \ldots, \mathrm{T}\} \tag{4.16}$$

$$l_{xi} + l_{zi} \leq 1; \forall i \in \{1, \ldots, \mathrm{N}\} \tag{4.17}$$

$$l_{zi} + h_{zi} \leq 1; \forall i \in \{1, \ldots, \mathrm{N}\} \tag{4.18}$$

$$l_{zi} - w_{yi} + h_{zi} \leq 1, \forall i \in \{1, \ldots, \mathrm{N}\} \tag{4.19}$$

$$l_{zi} - w_{yi} + h_{zi} \geq 0, \forall i \in \{1, \ldots, \mathrm{N}\} \tag{4.20}$$

$$1 - l_{xi} - l_{zi} + w_{yi} - h_{zi} \leq 1, \forall i \in \{1, \ldots, \mathrm{N}\} \tag{4.21}$$

$$1 - l_{xi} - l_{zi} + w_{yi} - h_{zi} \geq 0, \forall i \in \{1, \ldots, \mathrm{N}\} \tag{4.22}$$

$$l_{xi} + l_{zi} - w_{yi} \leq 1, \forall i \in \{1, \ldots, \mathrm{N}\} \tag{4.23}$$

$$l_{xi} + l_{zi} - w_{yi} \geq 0, \forall i \in \{1, \ldots, \mathrm{N} \tag{4.24}$$

$$\sum_{i=1}^{\mathrm{N}} \mathrm{m}_i p_{ij} + \mathrm{C} + C_{sep} + C_{sp} \leq \mathrm{C}_{max}, \forall j \in \{1, \ldots, \mathrm{T}\} \tag{4.25}$$

where:

$$C_{sp} = \mathrm{t}\eta_{jk}\mathrm{H}_k(\rho_x \mathrm{W} + \rho_y \mathrm{L})$$

$$C_{sep} = \mathrm{t}\eta_{jk}\mathrm{H}_k(\rho_y \sum_{i=1}^{\mathrm{N}} (y'_i - y_i)p_{ij} + \rho_x \sum_{i=1}^{\mathrm{N}} (x'_i - x_i)p_{ij})$$

The problem (4.1) - (4.25) aims at minimizing the linear combination of the unoccupied volume inside the trays and the number of needed trays, respectively weighted by the constant terms α_1 and α_2, as defined in eq. (4.1).

Constraints (4.2) and (4.3) impose that, due to sectorisation, the coordinates x_i and y_i of the bottom left front corner of item i must be respectively equal to integer multiples of the pitches s_w and s_l. Then, constraints (4.4) and (4.5) respectively impose that each item can be placed in only one tray and that the height of each tray can assume only one height, among the available ones. Equations (4.6) impose that the variables f_i must be equal to the sum of the dimension of item i along x-axis, the grip space along the x-axis, and the thickness of the separator. Constraints (4.7)-(4.13) are the constraints necessary to model the relative position of the items and ensure that the items and separators do not overlap. In particular, constraints (4.7)-(4.8) model the relative positions between two items along the x-axis, i.e., if item i is on the left ($a_{ii'} = 1$) or on the right ($b_{ii'} = 1$) of item i' (see scenarios 4 and 5 of Fig. 4.4), the x-coordinates of the items must be assigned taking into account the dimension of the items, and also the gripping space, the separator size, and the sectorisation requirements. In fact, the variable D_i is introduced to satisfy the business rule imposing that the longitudinal dividers must extend throughout the length of the tray. In particular, the variable D_i is the sum of f_i and the eventual additional space necessary to place the i'-th item exactly in the next sector, without overlapping with the separator. Then, constraints (4.9) and (4.10) are the bounding constraints of variables D_i. The physical meaning of these constraints is that, if item i is placed exactly under item i' (i.e., $c_{ii'}$=1), or exactly over item i' (i.e., $d_{ii'}$=1), then D_i and $D_{i'}$ must be equal. The inequalities (4.11) set the value of $c_{ii'} = 1$ if item i is located in front of item i', and finally the inequalities (4.12) set the value of $d_{ii'} = 1$ if item i is located in behind of item i' (see Fig. 4.4 for all possible configurations of items' relative positioning). Then, constraints (4.13) ensure that if items i and i' are in the same tray they must have only on possible relative position.

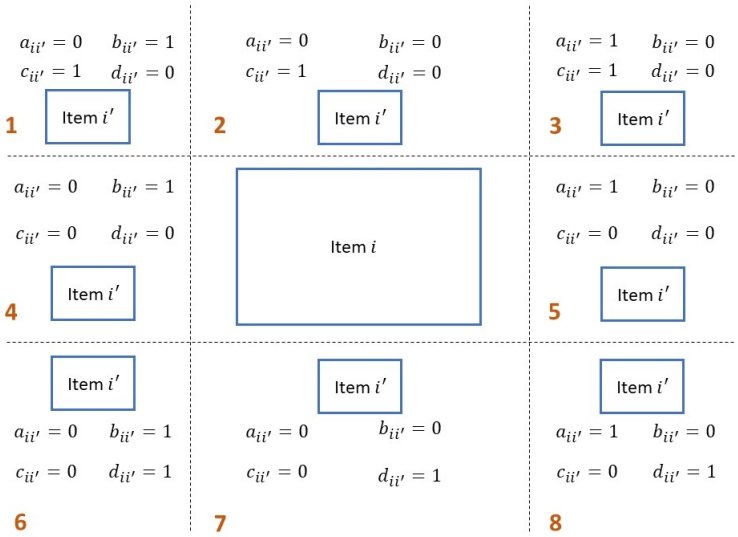


Figure 4.4: Possible relative positions between item i and item i' and variables assignment

Constraints (4.14)-(4.16) are geometric constraints, whose aim is ensuring that each item is placed completely inside the corresponding tray. Their formulation is similar to the ones

presented in [6], which has been adapted for this particular VLM use. In particular, inequality (4.14) determines the position along the x-axis of each item i and takes into account also the space requested by the longitudinal dividers, while (4.15) and (4.16) determine the position of the item along the y and z axes taking into account item's rotation and the grip space.

Constraints (4.17) - (4.24) are the consistency constraints that impose the right binary value to the variables according to the rotation and the position of the item. Finally, constraints (4.25) are the weight constraints that ensure the fulfillment of the weight limits for each tray. The inequality takes into account the total weight of the inserted items ($\sum_{i=1}^{N} \mathrm{m}_i p_{ij}$), the gross weight of the tray's base C, the weight of the left and bottom borders C_{sp}, and the weight of the dividers and the upper and right borders C_{sep}. These values are obtained considering the tare, calculated starting from the sum of the gross weight of the empty tray, the weight of the borders, and the weight of the dividers. Assuming that the volume density is the same for the borders and dividers, parameters ρ_x and ρ_y, corresponding to the volume density along the two axes, are introduced. Moreover, it is highlighted that it is important to note that there is no direct information available on the coordinates of the separators, as they are taken into account during the placement of items.

4.4.1.1 Model Linearization

The optimization problem described above is a MINLP due to the presence of some non-linear constraints, which contain some cross-products between decision variables. In particular, a non-linearity is present in constraints (4.10) and (4.25) and can be linearized following the method described in [7], by adding for each cross-product one auxiliary variable and four additional constraints for each auxiliary variable. The first two cross-products are present in the left side of equation (4.10):

$$d_{i'}(1 - a_{ii'} - b_{ii'})$$

and can be written as:

$$d_{i'} - d_{i'}a_{ii'} - d_{i'}b_{ii'} = d_{i'} - \Delta^1_{ii'} - \Delta^2_{ii'}$$

where $\Delta^1_{ii'}$ is equal to $d_{i'}a_{ii'}$ $\forall i, i' \in \{1, \ldots, \mathrm{N}\}, i \neq i'$ and is subject to following constraints:

$$\Delta^1_{ii'} \leq \mathrm{L}(a_{ii'}) \tag{4.26}$$

$$\Delta^1_{ii'} \geq 0 \tag{4.27}$$

$$\Delta^1_{ii'} \leq d_{i'} \tag{4.28}$$

$$\Delta^1_{ii'} \geq d_{i'} - \mathrm{L}(1 - a_{ii'}) \tag{4.29}$$

and $\Delta^2_{ii'}$ is equal to the $d_{i'}b_{ii'}$ $\forall i, i' \in \{1, \ldots, \mathrm{N}\}, i \neq i'$ and is subject to the following constraints:

$$\Delta^2_{ii'} \leq \mathrm{L}(b_{ii'}) \tag{4.30}$$

$$\Delta^2_{ii'} \geq 0 \tag{4.31}$$

$$\Delta^2_{ii'} \leq d_{i'} \tag{4.32}$$

$$\Delta^2_{ii'} \geq d_{i'} - \mathrm{L}(1 - b_{ii'}) \tag{4.33}$$

The other two cross-products are present in the left side of equation (4.25):

$$C_{sep} = \mathrm{t}\eta_{jk}\mathrm{H}_k(\rho_x \sum_{i=1}^{N}(y'_i - y_i)p_{ij} + \rho_y \sum_{i=1}^{N}(x'_i - x_i)p_{ij})$$

where it is assumed, due to sectorialization:

$$(x'_i - x_i) = D_i$$
$$(y'_i - y_i) = \mathrm{l}_i l_{yi} + \mathrm{w}_i w_{yi} + \mathrm{g}_i(1 - \sigma_i) + \mathrm{t})$$

which can be written using two new auxiliary variables $\Gamma^1_{ij'}$ and $\Gamma^2_{ij'}$:

$$\rho_x \sum_{i=1}^{N}(y'_i - y_i)p_{ij} = \rho_x \sum_{i=1}^{N} \Gamma^1_{ij'}$$
$$\rho_y \sum_{i=1}^{N}(x'_i - x_i)p_{ij} = \rho_y \sum_{i=1}^{N} \Gamma^2_{ij'}$$

where $\Gamma^1_{ij'} = D_i p_{ij}, \forall i \in \{1, \ldots, \mathrm{N}\}, \forall j \in \{1, \ldots, \mathrm{T}\}$ subject to:

$$\Gamma^1_{ij'} \leq \mathrm{L}p_{ij} \tag{4.34}$$
$$\Gamma^1_{ij'} \geq 0 \tag{4.35}$$
$$\Gamma^1_{ij'} \leq D_i \tag{4.36}$$
$$\Gamma^1_{ij'} \geq D_i - \mathrm{L}(1 - p_{ij}) \tag{4.37}$$

where $\Gamma^2_{ii'} = (\mathrm{l}_i l_{yi} + \mathrm{w}_i w_{yi} + \mathrm{g}_i(1 - \sigma_i) + \mathrm{t})p_{ij}, \forall i \in \{1, \ldots, \mathrm{N}\}, \forall j \in \{1, \ldots, \mathrm{T}\}$ subject to:

$$\Gamma^2_{ii'} \leq \mathrm{W}p_{ij} \tag{4.38}$$
$$\Gamma^2_{ii'} \geq 0 \tag{4.39}$$
$$\Gamma^2_{ii'} \leq \mathrm{l}_i l_{yi} + \mathrm{w}_i w_{yi} + \mathrm{g}_i(1 - \sigma_i) + \mathrm{t} \tag{4.40}$$
$$\Gamma^2_{ii'} \geq \mathrm{l}_i l_{yi} + \mathrm{w}_i w_{yi} + \mathrm{g}_i(1 - \sigma_i) + \mathrm{t} - \mathrm{W}(1 - p_{ij}) \tag{4.41}$$

The final model is composed by the objective function:

$$\min[\alpha_1 \ \mathrm{WL}_k(\sum_{j=1}^{T}\sum_{k=1}^{K}\eta_{jk}\mathrm{H}_k) + \alpha_2(\sum_{j=1}^{T}\sum_{k=1}^{K}\eta_{jk})] \tag{4.42}$$

Subject to:
(4.2)-(4.9), (4.17)-(4.24), (4.26)-(4.41)

$$D_i \geq d_{i'}(1 - \Delta^1_{ii'} - \Delta^2_{ii'}), \forall i, i' \in \{1, \ldots, \mathrm{N}\}, i \neq i' \tag{4.43}$$

$$\sum_{i=1}^{N} \mathrm{m}_i p_{ij} + \mathrm{C} + \mathrm{t}\eta_{jk}\mathrm{H}_k(\rho_x \sum_{i=1}^{N} \Gamma^1_{ij'} + \rho_y \sum_{i=1}^{N} \Gamma^2_{ij'}) +$$
$$C_{sp} \leq \mathrm{C}_{max}, \forall j \in \{1, \ldots, \mathrm{T}\} \tag{4.44}$$

4.4.2 Trays' Allocation Problems

In this subsection, it is presented the two MILP models for the allocation of the trays inside the VLM columns, both performed by optimizing one column at a time. As defined by the business rules and the requirements of the problems, the input parameters required for solving the problems are shown in Table 4.4, whereas the variables are shown in Table 4.5.

Table 4.4: Trays' allocation parameters

Parameter	Description
$B_{kk'}$	Normalized difference between the r and r' types of trays to be inserted in the columns
Z_k	Total quantity of the k-th type of tray
Z'_k	Total quantity of the k-th type of tray inserted at each iteration
O	Height of the column
O_f	Height of the free space in the column

Table 4.5: Trays' allocation variables

Name	Description
r	Index of the r-th tray type
z_{1r}	Total quantity of the r-th full tray type in the column
z_{2r}	Total quantity of the r-th empty tray type in the column
γ	Auxiliary binary variable

4.4.2.1 Filled Trays' Allocation Problem

The objective function considers maximizing the space occupied by the trays in the column.

$$\max \sum_{r=1}^{R} z_{1r} H_r \tag{4.45}$$

subject to:

$$\sum_{r=1}^{R} z_{1r} H_r \leq O \tag{4.46}$$

$$z_{1r} + Z'_r \leq Z_r \forall r \in \{1, ..., R\} \tag{4.47}$$

$$z_{1r} - z_{1r'} - z_{1r}(1 + B_{rr'}) \leq 0 \quad \forall r \in \{2, ..., R\}, \; r' \in \{1, ..., r-1\}, r' < r \tag{4.48}$$

$$\sum_{r=1}^{R} z_{1r} > 0 \tag{4.49}$$

where

$$B_{rr'} = \frac{(Z_r - Z'_r) - (Z_{r'} - Z'_{r'})}{|(Z_r - Z'_r) - (Z_{r'} - Z'_{r'})|}$$

In particular, constraint (4.46) is the geometric constraint that ensures that the configuration of the filled trays fits in the column. Constraints (4.47) are the overflow constraints that ensure that the quantity of trays inserted in the column is consistent with the total quantity given as input for each type of tray, whereas constraints (4.48) and (4.49) are the priority constraints: the former ensures that the type with the highest priority gets more trays inside the column. This is based on the ordering of trays according the given priority rule; while the latter imposes that the quantity of the trays of at most one category must be a positive quantity, i.e., there must not be empty columns.

4.4.2.2 Empty Trays' Allocation Problem

The objective function considers maximizing the space occupied by the trays in the column.

$$\max \sum_{r=1}^{\mathrm{R}} z_{2r}\mathrm{H}_r \tag{4.50}$$

subject to:

$$\sum_{r=1}^{\mathrm{R}} z_{2r} \cdot \mathrm{H}_r \leq \mathrm{O_f} \tag{4.51}$$

$$\sum_{r=1}^{\mathrm{R}} z_{2r} > 0 \tag{4.52}$$

$$\frac{z_{2r}}{\sum_{r=1}^{\mathrm{R}} z_{2r}} \leq \frac{\mathrm{Z}_r}{\mathrm{R}} + \mathrm{M}\gamma \quad \forall r \in \{1, ..., \mathrm{R}\} \tag{4.53}$$

In particular, constraint (4.51) is the geometric constraint that ensures that the configuration of the empty trays fits in the column. Constraints (4.52) and (4.53) are the frequency constraints: the former ensures that the quantity of the specific type of trays is not null, while the latter ensures that the frequency of empty trays of each type is at most equal to the frequency of full trays in the column. Note that the presence of M and γ is used for evaluating the eligible region even if this constraint is not satisfied. Moreover, constraint (4.53) is non-linear and so it is substituted with the following linear constraint:

$$z_{2r} - \frac{\mathrm{Z}_r}{\mathrm{R}} \cdot \sum_{r=1}^{\mathrm{R}} z_{2r} \leq \mathrm{M}\gamma \quad \forall r \in \{1, ..., \mathrm{R}\} \tag{4.54}$$

The final model contains equations (4.50)-(4.52) and (4.54).

4.5 Experimental Results

This section shows the validation of the proposed algorithm through a wide set of tests conducted implementing the matheuristics with Matlab2021b, on a PC equipped with an Intel(R) Core(TM) i7-10510U CPU and 12 GB of RAM in Windows 11. Since the applications of VLM are still not extensively discussed in the literature and to the best of available knowledge there are no reference datasets available, the algorithm is tested considering realistic datasets provided by an Italian logistic company.

4.5.1 Test of the Trays Internal Configuration Phase

To test the validity of the first part of the algorithm, described in Subsection 4.3.1, the following dataset is considered. In particular, the parameters of the model are set as follows:

- Tray's size
  - L = 800mm;
  - W = 600mm;
  - H = 475mm;

- Maximum weight supported by a tray C_{max}=120kg:
- Pitch of the dividers along the width and the length of the tray:
  - s_w = 50mm;
  - s_l = 100mm;
- Weight of the first/second term of the objective function α_1=α_2=0.50;
- Arbitrarily large number: M = 10000;
- Dividers thickness: t = 20mm;
- Gripping space: g_i = 20mm;
- Volume density of the dividers and edges: $\rho_x = \rho_y = 0.0027 g/mm^3$;
- Maximum running time: MaxTime = 7200s;
- Maximum number of patterns required by the company P = 32;

The items' identification number (ID), mass, geometric dimensions, and quantities are reported in Table 4.6.

Table 4.6: Items' features

ID	m [g]	l [mm]	w [mm]	h [mm]	N
1	150	40	100	13	1500
2	250	62	138	14	1500
3	1500	50	50	240	30
4	1500	40	40	180	100
5	1500	90	90	400	30
6	580	80	125	245	50
7	30	250	120	20	100
8	500	68	68	240	300
9	1500	125	125	100	30
10	200	125	125	10	30
11	300	230	230	100	20
12	300	230	230	10	20
13	1500	160	160	90	30
14	1500	250	130	230	40
15	130	300	340	20	130
16	500	90	90	40	50
17	500	180	180	40	30
18	500	90	90	105	50
19	500	190	190	110	20
20	500	290	290	40	20
21	20	170	170	40	30
22	1500	300	300	200	10
23	700	120	120	100	20
24	1000	140	280	150	40
25	100	50	400	50	40

As previously explained, the first sub-phase of the *trays internal configuration phase* deals with the allocation of items with the same ID into mono-item trays. In particular, after the

grouping of the data by ID, the algorithm starts by checking if there are some input items whose dimensions do not fit in the VLM: if they exceed the highest trays' available height, or have some dimensions that do not fit the machine parameters they are discarded. Then, if some items exceed the maximum tray splitting width they are considered for the definition of a tray containing a single item, otherwise for each of the remaining items, the side of the item with the widest area is used for calculating the approximate number N_t of items that can be contained within the mono-item tray. Table 4.7 shows the obtained results, specifically, the first column represents the item ID code, the second shows the quantity of the items (also reported in Table 4.6), the third represents the potential capacity of the tray, thus the number of items contained within a mono-item tray (N_t), the fourth represents the number of mono-item trays filled, and finally, the fifth column reports the number of remaining items that will be considered in the mixed-items trays internal configuration problem. According to the achieved results, the configurations of mono-item trays created are 21 and the number of total filled trays is 91. Figure 4.5 reports 3

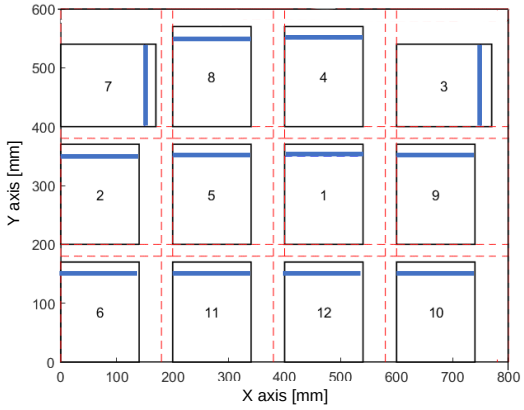
Table 4.7: Outcomes of the mono-item trays sub-phase

ID	N_c	Filled trays	Residual Items
1	96	15	60
2	48	31	12
3	48	0	30
4	48	2	4
5	16	1	14
6	24	2	2
7	48	2	4
8	32	9	12
9	16	1	14
10	48	0	30
11	8	2	4
12	24	0	20
13	16	1	14
14	8	5	0
15	24	5	10
16	32	1	18
17	24	1	6
18	16	3	2
19	8	2	4
20	12	1	8
21	24	1	6
22	4	2	2
23	16	1	4
24	12	3	4
25	48	0	40

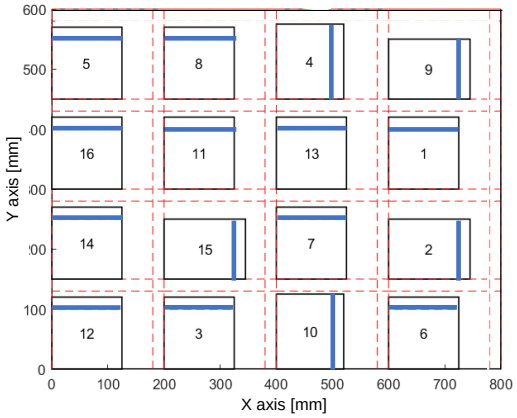
examples of filled mono-item trays, the first shows a tray with 12 items with ID 24, the second shows a tray with 16 items with ID 9, and finally, the third shows a tray with 24 items with ID 6. As it can be seen in Figure 4.5a and Figure 4.5b, the solver succeeds in allocating all the items within the tray, as opposed to Figure 4.5c in which the solver, given the high computational complexity, allocates 20 over 24 items.

Proceeding with the test of the *trays internal configuration phase*, the second sub-phase receives as input the list of residual items not assigned to the mono-item trays (corresponding

(a) *Item 24*



(b) *Item 9*



(c) *Item 6*

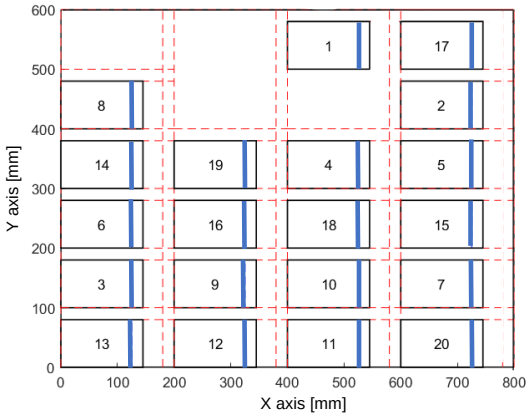


Figure 4.5: Example of single-type trays

to the fourth column in Table 4.7) and aims first at clustering the residual items, as explained in Section 4.3.2. The number of clusters is calculated as the difference between the number of patterns desired by the company (i.e., 32) and the number of mono-item tray configurations created during the first sub-phase (i.e., 21). As in the previous step, first, it is calculated the number of items per cluster that can be contained within the mixed trays (considering the maximum dimension of the items contained in the cluster) while respecting the physical limits of tray subdivision and its sustainable weight limits. Then the mathematical model is executed. Also in this case, it can happen that the number of assigned items is lower than the firstly computed one. For this reason, this sub-phase is executed iteratively, updating at each iteration the list of residual items until all items are assigned. For this dataset, the clusterization is performed three times and the obtained results are summarized in Table 4.8 where the first column represents the clusters' ID, the second column represents the number of items assigned to the corresponding cluster, the third column the number of items of the cluster assigned to the tray, the fourth column reports the number of filled trays, and finally the fifth column reports the number of residual items of the cluster. As it can be evicted from the table, the number of clusters created

at each iteration with the *k-means* algorithm (performed using width and length as grouping parameters) changes at each clusterization. In the first clusterization, 11 clusters are considered. However, after the solution of the item assignments MILP problem, only 7 configurations are defined (see IV column in Table 4.8). Consequently, in the second clusterization, 4 clusters are initially considered. The same holds for the third clusterization.

Figure 4.6a shows the division into clusters, in which each "dot" corresponds to one of the 25 IDs of the items while Fig. 4.6b shows how each color corresponds to the region of one cluster, containing different types of items. The axes of Fig. 4.6b are respectively the length and the width of the items, i.e., the parameters of the k-means algorithm.

Table 4.8: Resulting dataset after each clustering

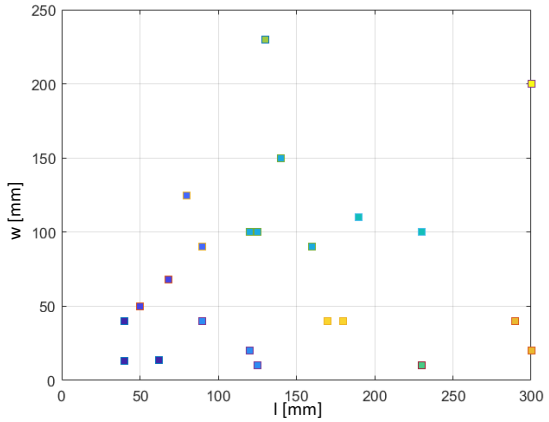
Cluster ID	N	N_c	Filled trays	Residual Items
First clusterization				
1	76	48	1	28
2	82	32	2	18
3	18	16	1	2
4	52	32	1	20
5	36	12	3	0
6	8	8	1	0
7	20	24	0	20
8	0	8	0	0
9	18	12	1	6
10	12	24	0	12
11	2	4	0	2
Second clusterization				
I	68	16	4	4
II	2	4	0	2
III	32	16	2	0
IV	6	12	0	6
Third clusterization				
A	12	6	2	12
B	0	0	0	0

As it can be evicted from Table 4.9 the first phase of the proposed matheuristics allows the computation of consistent results with respect to the initial requirements. In fact, with the combination of mathematical programming and heuristics, it is possible to obtain a short computation time, which would have not been possible with just an exact resolution of the problem.

Table 4.9: Outcomes of the trays internal configuration phase

	Patterns	Filled Trays	Comp. Time [s]
Mono-item trays	21	91	0.9
First clustering mixed trays	7	10	0.15
Second clustering mixed trays	4	8	30.43
Third clustering mixed trays	2	2	30.37
Total	32	109	31.46

(a) *First clusterization clusters - The dots correspond to the one of the 25 different input IDs of the items*



(b) *Section of the first clusterization clusters*

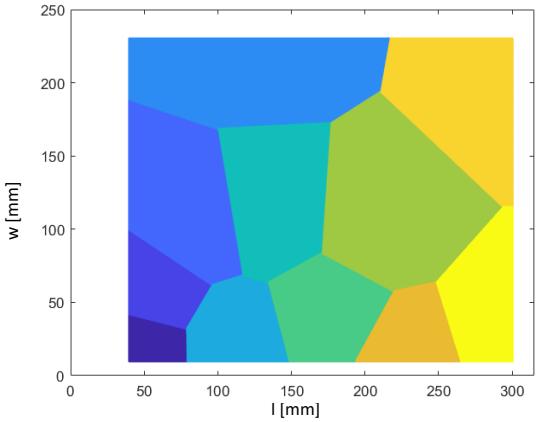


Figure 4.6: Clusters

With the aim of evaluating the performance of this phase of the algorithm, the average fill ratio is computed. This metric computes the percentage of tray filling calculated on average over the total number of trays, the results are as follows:

1. **Mono-item trays**: The average fill ratio over the total of the 91 mono-item trays is of the 60%. This value is due to the imposition of constraints, especially those on dividers' placement. In fact, it is evident, especially in Figures 4.5a - 4.5b, the fill ratio can actually be considered to be 100% because the exact number of items contained within the trays was calculated considering all the business constraints;
2. **Mixed-items trays**: it is computed the fill ratio separately for the two groups of trays obtained with the two consecutive clusterings:
   - the fill ratio of the 10 mixed trays obtained from the first-level clustering turns out to be 71.5%;
   - the fill ratio of the 2 mixed trays obtained from the second-level cluster turns out to be 75.4%.

   It is important to point out that the reported fill ratio is calculated by considering, for each cluster, the maximum value of the geometric dimensions of the related items since the vertical dividers are placed according to the dimension of the largest item placed in that column, thus leading to a waste of space if there are items with a smaller dimension, consequently, it is rounded up.

4.5.2 Trays' Allocation

Based on company's most frequent warehouses' configurations, for the test of the second phase of the matheuristics, 36 different evaluation scenarios have been created as the combination of 3 different columns' heights O_1=3000 mm, O_2=9000 mm, O_3=9000 mm, 4 different trays' types whose heights are H_1=75 mm H_2=125 mm, H_3=225 mm, and H_4=325 mm and whose total quantity T and the quantity for each type of tray Z_r range from 12 to 48, so that the scenarios

Table 4.10: Priority Rules Description

Priority Rule	Name	Description
A	Largest height first	Choose the tray with the largest height
B	Largest height with increasing frequency	Choose the largest height with increasing frequency, i.e., first choose the largest height r with $\varphi_{1r} \geq 1$. Then choose the largest height r' with $\varphi_{1r'} \geq \varphi_{1r}$.
C	Largest height with pseudo-increasing frequency (first version)	Choose the largest height with pseudo-increasing frequency, i.e., first choose the largest height r with $\varphi_{1r} \geq 1$. Then choose the largest height r' with $\varphi_{1r'} \geq 2$. The ith chosen value should have frequency $\varphi_{1r} \geq i$
D	Largest height with pseudo-increasing frequency (second version)	Choose the largest height with pseudo-increasing frequency, i.e., first choose the largest height r with $\varphi_{1r} \geq 1$. Then the ith chosen value should have frequency $\varphi_{1r} \geq 4i/\mathrm{T}$
E	Most frequent height	Choose the most frequent height
F	Largest height with increasing frequency and threshold on the frequency	It is equivalent to priority rule (B) but if less than Φ heights are selected, priority rule (A) is applied for the remaining heights to be chosen
G	Largest height with pseudo-increasing frequency and most frequent height	It is equivalent to priority rule (C) but if less than Φ heights are selected, priority rule (A) is applied for the remaining heights to be chosen
H	Largest height with pseudo-increasing frequency and threshold on the frequency (first version)	It is equivalent to priority rule (C) but if less than Φ heights are selected, priority rule (E) is applied for the remaining heights to be chosen
I	Largest height with pseudo-increasing frequency and threshold on the frequency (second version)	It is equivalent to priority rule (D) but if less than Φ heights are selected, priority rule (E) is applied for the remaining heights to be chosen
L	Decimal and less frequent height first	Choose the trays with decimal height and ordered ascendingly by frequency, then choose the largest integer height.
M	Integer height and the largest size first	Choose the trays with integer height and ordered decreasingly by height then choose the trays with decimal height and ordered ascendingly by frequency
N	Trays with the most frequent ratio $\mathrm{O/H}_j$ first	Given the ratio of column height to the height of tray $\mathrm{O/H}_j$ choose the most frequent trays with the higher ratio. Then give priority to the trays with the largest height.
O	Trays with the highest ratio $\mathrm{O/H}_j$ first	Given the ratio of column height to the height of tray $\mathrm{O/H}_j$ choose the trays with the highest values of the ratio. Then give priority to the trays with the most frequent height.
P	Largest height whose quantity of trays is more than Z_r is higher than $\sum^r(\mathrm{Z'}_r)$	Choose the trays with the largest height. Then choose the largest height whose quantity Z_r i s higher than the average of the quantity of all the trays to be inserted

differ one from the other by their size and degree of heterogeneity. Particular attention in this phase is given to the definition of the priority rules since their use can significantly improve the performance of the whole algorithm and are crucial for its scalability in the different types of implementation.

4.5.2.1 Priority Rules

The priority rules are defined as preference criteria for the ordering of the trays in each column of the VLM. In particular, they are used to determine the order in which the tray should be positioned. In general, they allow finding a trade-off between conflicting objectives, such as giving higher priority to the trays with the largest dimensions or giving higher priority to the trays whose dimensions more frequently recur in the dataset. Here, two different types of priority rules are considered: the first type is derived from the work on the container loading problem presented by Pisinger [8], and then adapted to this one-dimensional problem (it is only considered the height of the tray); the latter is defined in accordance with the industrial needs. In the following the list of priority rules is shown, while for each priority rule the detailed explanation can be found in Table 4.10. In this case study, given the frequency φ_{1r} of the generic tray of type r and a generic integer number Φ used as threshold for selecting a predefined number of heights, the priority rules taken from [8] and adapted for this application are the following:

A: Largest height first;

B: Largest height with increasing frequency;

C: Largest height with pseudo-increasing frequency (first version)

D: Largest height with pseudo-increasing frequency (second version)

E: Most frequent height;

F: Largest height with increasing frequency and threshold on the frequency (combination of A and B);

G: Largest height with pseudo-increasing frequency and most frequent height (combination of A and C);

H: Largest height with pseudo-increasing frequency and threshold on the frequency (first version) (combination of C and E);

I: Largest height with pseudo-increasing frequency and threshold on the frequency (second version) (combination of D and E);

The newly implemented priority rules are:

L: Decimal and less frequent height first;

M: Integer height and the largest height first;

N: Trays with the most frequent ratio O/H_j first;

O: Trays with the highest ratio O/H_j first;

P: Largest height whose quantity of trays Z_r is higher than $\sum^r(Z'_r)$.

4.5.2.2 Test

The graphs and tables below show the results obtained from the simulations with columns of heights equal to O_1= 3000mm, O_2=9000mm, and O_3=15000mm, four different types of trays indicated by R_1, R_2, R_3 and R_4 characterized by combinations of different heights values, i.e., H_1=75mm, H_2=125mm, H_3=225mm, H_4=325mm, and their total quantity that can assume the values T_1=12, T_2=24, T_3=48. The combination of O, R, and T produces 36 different test scenarios, that for the sake of clarity in the following will be identified with the letter and the number of each of the three parameters, e.g., $O_1R_2T_1$ indicates the scenario with the height of the column equal to 3000mm, with 12 trays with 2 different heights.

For the evaluation of this phase it is considered the number of columns obtained with the algorithm, the corresponding fill ratio, and the computation time. Evidently, depending on the priority rule used for each scenario, the results can largely vary. Table 4.11 and Table 4.12 show respectively the worst and the best performance achieved for each scenario, highlighting the corresponding priority rule. More in detail, for each scenario, the two tables report the number of filled columns, the number and the type of empty trays used, the percentage of occupied space, and the execution time. As expected, the datasets composed of 4 types of trays, are the ones with the most critical computation time. The worst performance does not depend on a particular business rule since there is not any frequent business rule or any correlation with the type of column and/or type of trays. Differently, the best performace reported in Table 4.12 show that priority rule A is the most frequent one among the various cases. For this priority rule the minimum percentage of space occupied by full trays is 20%, for the 15000 mm column considering a small homogeneous scenario with one type of tray, and for the majority of cases, it is greater than 40%, reaching 100% for the 3000 mm column considering a medium homogeneous scenario with 3 types of full trays. From the achieved results, it can be evicted that the execution time is mainly influenced by three main factors:

1. The number of tray types in input;
2. The column height;
3. The remaining free space generated by the allocation of full trays in the columns, used to determine the number of empty trays.

Summarizing the obtained results, it can be stated that an occupancy rate of approximately 100 % is achieved with the use of empty trays. The lowest occupancy rate achieved is approximately 97 % for scenarios $O_2R_1T_1$ and $O_2R_1T_3$. As for the number of filled columns, the best result

Table 4.11: Worst cases performance indices

Input					Output			Performance indices		
O (mm)	**R**	**T**	**Input Trays**		z_{2r}	**Priority Rule**	**Number of columns**	**Comp. Time (sec)**	**Average fill ratio (full trays only) %**	**Average fill ratio (empty trays added) %**
			H (mm)	Z_r						
3000		12	75	12	18	F	1	0.016	40	100
	1	24	75	24	6	B	1	0.016	80	100
		48	75	48	12	A	2	0.078	80	100
		12	125; 75	5; 7	9; 2	M	1	0.024	48.3333	99.9667
	2	24	325; 75	12; 12	0; 6	F	2	0.032	90	99.9833
		48	325; 75	24; 24	0; 12	A	4	0.078	90	100
		12	325; 225; 75	2; 4; 6	2; 0; 0	P	1	0.063	76.6667	99.9667
	3	24	225; 125; 75	8; 8; 8	0; 4; 14	P	2	0.134	66.6667	100
		48	225; 125; 75	8; 16; 24	0; 0; 22	C	3	0.140	75.5556	100
		12	325; 225; 125; 75	1; 2; 4; 5	3; 0; 0; 0	H	1	0.047	65	99.9667
	4	24	325; 225; 125; 75	2; 4; 8; 10	0; 0; 0; 21	P	2	0.203	65	100
		48	325; 225; 125; 75	12; 12; 12; 12	0; 0; 0; 18	L	4	0.234	85	100
9000	1	12	75	12	78	D	1	0.016	13.3333	100
		24	75	24	66	A	1	0.016	26.6667	100
		48	225	48	24	M	2	0.018	66.6667	100
		12	125; 75	6; 6	30; 30	C	1	0.125	16.6667	100
	2	24	125; 75	12; 12	24; 24	N	1	0.078	33.3333	100
		48	225; 125	24; 24	0; 56	L	2	0.070	53.3333	100
		12	225; 125; 75	2; 4; 6	18; 12; 10	L	1	1.720	18.8889	99.9889
	3	24	225; 125; 75	4; 8; 12	20; 4; 0	O	1	0.913	37.7778	99.9889
		48	325; 125; 75	16; 16; 16	0; 0; 84	M	2	1.171	53.3333	100
		12	325; 225; 125; 75	1; 2; 4; 5	14; 4; 7; 1	I	1	11.747	21.6667	99.9889
	4	24	325; 225; 125; 75	2; 4; 8; 10	13; 1; 2; 0	M	1	3.468	43.3333	99.9889
		48	325; 225; 125; 75	12; 12; 12; 12	0; 0; 0; 78	A	2	22.423	56.6667	100
15000	1	12	75	12	138	A	1	0.025	8	100
		24	125	24	76	F	1	0.016	24	100
		48	75	48	102	A	1	0.031	32	100
		12	125; 75	5; 7	55; 53	D	1	0.219	9.6667	99.9933
	2	24	125; 75	12; 12	48; 48	M	1	0.219	20	100
		48	125; 75	24; 24	36; 36	I	1	0.141	40	100
		12	225; 125; 75	2; 4; 6	25; 27; 30	L	1	9.647	11.3333	99.9933
	3	24	225; 125; 75	4; 8; 12	33; 19; 5	A	1	6.27	22.6667	99.9933
		48	225; 125; 75	8; 16; 24	31; 3; 0	I	1	2.407	45.3333	99.9933
		12	325; 225; 125; 75	1; 2; 4; 5	17; 10; 20; 16	P	1	110.635	13	99.9933
	4	24	325; 225; 125; 75	2; 4; 8; 10	26; 5; 5; 0	F	1	185.097	26	99.9933
		48	325; 225; 125; 75	4; 8; 16; 20	18; 3; 1; 0	G	1	12.560	52	99.9933

achieved for the majority of the scenarios is 1, while the worst is 4 obtained in scenarios $O_1R_2T_3$ and $O_1R_4T_3$. Table 4.13 shows the different average percentages of free space and occupied space left at the end of the processing sub-phase (i.e., with the insertion of only full trays) and the end of the post-processing sub-phase (i.e., the complete columns with both empty and full trays) and the computation time, all grouped by the type of used dataset (homogeneous if the total quantity of trays T has been divided equally among the various height values, heterogeneous otherwise). It can be noted that as the height of the column increases, on average, the free space produced by the allocation of full trays increases; furthermore, as the height of the column increases, the average execution time also increases. Moreover, it can be evicted that a homogeneous scenario, on average, performs better than a heterogeneous scenario for the average space occupied by full trays. From the execution time point of view, it ranges from a minimum of some milli-seconds for scenarios with column O_1, to a maximum of 17.8 seconds for scenarios with column O_3.

4.6 Conclusion

The optimization of the configuration of the Vertical Lift Modules (VLM) warehouses concerned both the computation of the optimal configuration for the allocation of items and the allocation of the trays in columns, which strongly influences its performance, as it allows the minimization of the unused space in the warehouse itself. This work proposes a novel matheuristic algorithm

Table 4.12: Best cases performance indices

Input					Output			Performance indices		
O (mm)	**R**	**T**	**Input Trays**		$\mathbf{z}_{2r}$	**Priority Rule**	**Number of columns**	**Comp. Time (sec)**	**Average fill ratio (full trays only)** %	**Average fill ratio (empty trays added) %**
			H (mm)	$\mathbf{Z}_r$						
3000		12	125	12	8	A	1	0.001	60	100
	1	24	325	24	0	A	3	0.001	93.3333	0
		48	125	48	12	A	3	0.001	80	100
		12	225; 75	6; 6	2; 4	A	1	0.001	70	99.9667
	2	24	325; 75	12; 12	0; 6	A	2	0.001	90	99.9833
		48	225; 75	24; 24	0; 6	B	3	0.001	93.3333	100
		12	325; 225; 125	4; 4; 4	0; 0; 0	A	1	0.001	100	0
	3	24	325; 225; 75	8; 8; 8	0; 0; 4	B	2	0.001	93.3333	99.9833
		48	325; 125; 75	8; 16; 24	0; 0; 14	D	3	0.015	84.4444	100
		12	325; 225; 125; 75	3; 3; 3; 3	0; 0; 3; 0	E	1	0.001	85	99.9667
	4	24	325; 225; 125; 75	6; 6; 6; 6	0; 3; 1; 0	A	2	0.031	85	99.9833
		48	325; 225; 125; 75	4; 8; 16; 20	0; 0; 0; 12	M	3	0.062	86.6667	100
9000		12	325	12	13	A	1	0.001	46.6667	97.2222
	1	24	225	24	12	A	1	0.001	66.6667	100
		48	325	48	2	A	2	0.001	93.3333	97.2222
		12	325; 225	6; 6	9; 9	A	1	0.010	40	100
	2	24	325; 225	12; 12	3; 3	A	1	0.001	80	100
		48	225; 75	24; 24	2; 1	D	1	0.001	93.3333	99.9889
		12	325; 225; 75	4; 4; 4	17; 1; 0	D	1	0.266	31.1111	99.9889
	3	24	325; 225; 125	8; 8; 8	4; 4; 4	A	1	0.050	66.6667	100
		48	325; 125; 75	8; 16; 24	4; 0; 0	A	1	0.030	84.4444	99.9889
		12	325; 225; 125; 75	3; 3; 3; 3	14; 5; 2; 0	P	1	8.202	28.3333	99.9889
	4	24	325; 225; 125; 75	6; 6; 6; 6	9; 3; 0; 0	D	1	1.297	56.6667	99.9889
		48	325; 225; 125; 75	4; 8; 16; 20	3; 0; 1; 0	F	1	0.14	86.6667	99.9889
15000		12	225	12	48	A	1	0.001	20	100
	1	24	325	24	18	A	1	0.001	56	98
		48	325	48	36	A	2	0.001	56	98
		12	325; 225	6; 6	19; 19	B	1	0.030	24	100
	2	24	325; 225	12; 12	13; 13	A	1	0.020	48	100
		48	325; 75	24; 24	12; 0	F	1	0.001	72	99.9933
		12	325; 225; 75	4; 4; 4	25; 13; 2	F	1	1.828	18.6667	99.9933
	3	24	325; 225; 125	8; 8; 8	12; 12; 12	I	1	0.859	40	100
		48	325; 225; 125	16; 16; 16	4; 4; 4	B	1	0.099	80	100
		12	325; 225; 125; 75	3; 3; 3; 3	23; 11; 11; 0	F	1	73.250	17	99.9933
	4	24	325; 225; 125; 75	6; 6; 6; 6	24; 6; 0; 0	I	1	63.464	34	99.9933
		48	325; 225; 125; 75	12; 12; 12; 12	13; 1; 0; 0	A	1	2.925	68	99.9933

Table 4.13: Average filling results

O (mm)	Homogeneous					Heterogeneous				
	Average free space (full trays only) %	Average free space (empty trays added) %	Average occupied space (full trays only) %	Average occupied space (empty tray added) %	Average Execution time (sec.)	Average free space (full trays only)%	Average free space (empty trays added) %	Average occupied space (full trays only) %	Average occupied space (empty trays added) %	Average Execution time (sec.)
3000	21.0375	0.56018	83.9384	99.5614	0.019667	23.4522	0.47943	77.9503	99.5671	0.02042
9000	48.454	0.59258	51.3876	99.6537	0.72521	51.2273	0.27664	48.6142	99.6438	0.39782
15000	61.4802	0.33755	38.3613	99.7062	3.8892	64.7417	0.17244	35.0998	99.7035	6.5418

that is composed of different mathematical models for the solution of this complex problem. The model is versatile as it can be applied with several innovative logistic constraints, fulfilling logistic companies' requirements and VLMs' features, such as the use of a limited number of tray's configurations (patterns) and the limitations on the placement of dividers inside trays. The effectiveness of the model is tested on various scenarios. Results show that the exact mathematical model turns out to be time-consuming for large instances, while with a limited number of items (and consequently variables) it produces optimal solutions in a short computation time. In order to extend the applicability of the model also to large datasets, future works will consider the

definition of heuristics able to overcome the emerged limitations and the use of other resolution approaches (i.e., genetic programming) to compare with the one just proposed.

References

[1] Tresca, G., Cavone, G., Carli, R., Cerviotti, A., and Dotoli, M., "Automating bin packing: A layer building matheuristics for cost effective logistics," *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 3, pp. 1599–1613, 2022.

[2] Koster, R. de, "Automated and robotic warehouses: Developments and research opportunities," *Logistics and Transport*, vol. 38, pp. 33–40, 2018.

[3] Dallari, F., Marchet, G., and Melacini, M., "Design of order picking system," *The international journal of advanced manufacturing technology*, vol. 42, no. 1-2, pp. 1–12, 2009.

[4] Dukic, G., Opetuk, T., and Lerher, T., "A throughput model for a dual-tray vertical lift module with a human order-picker," *International journal of production economics*, vol. 170, pp. 874–881, 2015.

[5] Sun, X., Wu, C.-C., and Chen, L.-R., "An automated warehouse sorting system for small manufacturing enterprise applying discrete event simulation," in *2018 2nd IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)*, IEEE, 2018, pp. 1597–1601.

[6] Chen, C., Lee, S.-M., and Shen, Q., "An analytical model for the container loading problem," *European Journal of operational research*, vol. 80, no. 1, pp. 68–76, 1995.

[7] Bemporad, A. and Morari, M., "Control of systems integrating logic, dynamics, and constraints," *Automatica*, vol. 35, no. 3, pp. 407–427, 1999.

[8] Pisinger, D., "Heuristics for the container loading problem," *European Journal of Operational Research*, vol. 141, no. 2, pp. 382–392, 2002.

Part 2: Delivery Planning and Online Replanning

Chapter 5

A MILP Approach for the Multi-Drop Container Loading Problem Resolution in Logistics 4.0

V

Abstract

This chapter addresses the multi-drop container loading problem (CLP), i.e., the problem of packing multiple bins, associated to multiple deliveries to one or more customers, into a finite number of transport units (TUs). Differently from the traditional CLP, the multi-drop CLP has been rarely handled in the literature, while effective algorithms to automatically solve this problem are needed to improve the efficiency and sustainability of internal logistics. To this aim, it has been proposed a novel algorithm that solves a delivery-based mixed integer linear programming (MILP) formulation of the problem. The algorithm efficiently determines the optimal composition of TUs by minimizing the unused space, while fulfilling a set of geometric and safety constraints, and complying with the delivery allocation. In particular, the proposed algorithm includes two steps: the first aims at clustering bins into groups to be compatibly loaded in various TUs; the latter aims at determining the optimal configuration of each group in the related TU. Finally, the proposed algorithm is applied to several realistic case studies with the aim of testing and analysing its effectiveness in producing stable and compact TU loading configurations in a short computation time, despite the high computational complexity of the multi-drop CLP.

Contents

5.1 Introduction

Taking advantage of the Industry 4.0 enabling technologies, the novel concept of Logistics 4.0 has been recently introduced, aiming at improving the productivity level and quality standards of demand, distribution, and inventory management [1], [2]. The new paradigms of mechanization, automation, and physical internet are leading to an enhanced management of the internal logistics, efficiently supporting the operative management of the physical flows that transit in the warehouse. Nevertheless, although a large variety of systems for the management of the internal logistics are available, effective container loading algorithms are still lacking. Such algorithms could support warehouse managers in automatically addressing the container loading problem (CLP):

the problem of packing multiple bins into a finite number of containers or transport units (TUs) [3]-[4].

CLP is a combinatorial optimization problem which belongs to the class of *cutting and packing* problems [5]. In the literature, the classical version of CLP is commonly described as finding the optimal packing of a set of bins, known as boxes, into containers, so that bins do not overlap and lie entirely inside the containers, while minimizing the non-occupied space. The resulting optimization problem thus focuses on the spatial arrangement of bins in the containers, considering only geometric constraints [3]. To make the proposed solutions relevant to real-world applications, several further variants of the CLP have been defined. A first broad classification of the literature contributions is based on the number of containers included in the optimization problem, thus distinguishing the single- and multi-CLP. Weight and static stability are commonly considered in the single-container case, while dynamic stability has received less attention to date [6]. In most cases, heuristic [7]–[9] and meta-heuristic [10]–[12] approaches have been followed, while mathematical models and exact algorithms have usually addressed only basic problems [13]–[16].

A second main classification of the literature contributions on the CLP can be made by considering the number of delivering destinations associated to the bins, thus distinguishing between mono- and multi-drop CLP [17]. For the sake of reducing the problem complexity, in the multi-drop CLP the route of delivering operations is commonly assumed to be known in advance. Thus, in the multi-drop scenario the resulting optimization problem consists in enhancing the CLP with a Last-In-First-Out (LIFO) policy aimed at fullfilling the loading and unloading of containers. Some heuristic [18], [19] and exact [17] algorithms have been proposed to arrange customer bins partitioned sections of the container. However, in the cited works [17]–[19] only one container is considered; moreover, all the constraints – except the geometric and the LIFO constraints – are generally neglected.

Specifically, the multi-drop scenario can be addressed in two ways. On the one hand, the CLP is integrated with the vehicle routing problem: this results in the combined optimization of the loading of bins into containers and the routing of containers along paths aimed at serving different destinations with the minimum traveling cost. Such an approach suffers from a high combinatorial complexity as shown in [20]–[22]. On the other hand, for the sake of reducing the problem complexity, the route of delivering operations is commonly assumed to be known in advance. Thus, in the multi-drop scenario the resulting optimization problem consists in enhancing the CLP with a Last-In-Fist-Out (LIFO) policy aimed at fullfilling the loading and unloading of containers. Finally, note that in Lai *et al.* [18], who propose a heuristic graph-based approach, and in Junqueira *et al.* [17], who formulate a mixed integer linear programming model for multi-customer delivering, only one container is considered; moreover, all the constraints (except the geometric and the multi-drop constraints) are neglected.

Summing up, CLP research is still lagging behind in jointly addressing the following type of constraints in a multi-drop scenario: load-balancing, multiple delivery, compliance with loading and unloading strategies, bins positioning including rotation and stackability, and bins static and dynamic stability. To cope with this gap, it has been proposed a novel mixed integer linear programming (MILP) delivery-based algorithm that efficiently determines the optimal composition of TUs, minimizing the unused space, fulfilling a set of geometric and safety constraints, and complying with the delivery allocation. In particular, it decomposes the resolution procedure into two steps: the first aims at clustering bins into groups to be compatibly loaded in various TUs; the latter aims at determining the optimal configuration of each group in the related TU. Finally, the proposed algorithm is applied to several real case studies, showing that the proposed technique produces stable and compact TU loading configurations within an acceptable time, despite the high computational complexity of the multi-drop CLP.

The remainder of this chapter is organized as follows. Section 5.2 describes the basic concepts

and assumptions and provides the mathematical formulation of the single-CLP with multi-drop constraints. Section 5.3 describes the algorithm for efficiently solving the multi-CLP in a multi-drop scenario. In Section 5.4 the proposed algorithm is applied to simulated case studies, showing the resulting performance. Finally, Section 5.5 provides the conclusions and some possible future research perspectives.

5.2 Mathematical Formulation of the Single-CLP with Multi-drop Constraints

The resolution method proposed in this work for the logistics multi-drop CLP takes as inputs a set of deliveries and their corresponding bins, and the type of TUs to be used for the shipment; while it returns as outputs the most efficient configurations of the input bins in TUs. All the bins related to a single customer are grouped into the so-called delivery. In effect, depending on the type of shipment, this can include one or more customer deliveries, while each delivery is associated only to a single customer. The deliveries associated to a given shipment can be loaded inside one or multiple TUs, depending on the type of TUs used for the shipment. A sufficient number of trucks are available to fully carry out the delivery of all the bins.

As for the composition of a single TU, the loading arrangement is determined in accordance with a Last-In-First-Out (LIFO) strategy. This strategy ensures that, when bins for multiple destinations are considered, the loading arrangement complies with the sequence by which each destination is served. It is also assumed that the arrangement of bins must be orthogonal (i.e., the edges of bins must be parallel to the edges of the TU). The TU capacity defines both the maximum weight and volume of bins that can be loaded. In addition, it is assumed that the bins' arrangement is done in accordance with the load balance (related to the position of the load center of gravity), which affects the maneuverability of the TU. Finally, it is assumed that the TU loading takes both the static (i.e., related to the capacity of the loaded bins to be in equilibrium when TU is stationary) and dynamic (i.e., related to the capacity of the loaded bins not to be displaced when the TU is moving) stability of bins into account.

In the rest of this section, the attention is focused on the mathematical formulation of the single-CLP problem with multi-drop constraints. The problem is here first formulated as a Mixed Integer Non-Linear Programming (MINLP) problem, which is then linearized by means of the approach proposed in [23]. For the sake of clarity, the decision variables and the parameters of the optimization problem are reported in Table 5.1 and Table 5.2, respectively.

The optimization problem aims at minimizing the unused space inside of the transport unit, i.e., the difference between the maximum volume capacity of the TU and the total volume of the bins assigned to the TU:

$$\min \left(\mathrm{V}_{\max} - \sum_{i \in \mathcal{B}} p_i \, \mathrm{v}_i \right) \tag{5.1}$$

while satisfying the following constraints:

1) Non-overlapping constraints: the coordinates of two generic bins i and i' within the TU must not overlap:

$$cx_i p_i - x_{i'} p_{i'} + \mathrm{W}\, \xi^{l}_{ii'} \leq \mathrm{W} - \mathrm{w}_i, \ \forall i, \forall i' \neq i \tag{5.2}$$

$$cx_{i'} p_{i'} - x_i p_i + \mathrm{W}\, \xi^{r}_{ii'} \leq \mathrm{W} - \mathrm{w}_{i'}, \ \forall i, \forall i' \neq i \tag{5.3}$$

$$cy_i p_i - y_{i'} p_{i'} + \mathrm{L}\, \xi^{b}_{ii'} \leq \mathrm{L} - \mathrm{l}_i, \ \forall i, \forall i' \neq i \tag{5.4}$$

$$cy_{i'} p_{i'} - y_i p_i + \mathrm{L}\, \xi^{f}_{ii'} \leq \mathrm{L} - \mathrm{l}_{i'}, \ \forall i, \forall i' \neq i \tag{5.5}$$

$$cz_i p_i - z_{i'} p_{i'} + \mathrm{H}\, \xi^{o}_{ii'} \leq \mathrm{H} - \mathrm{h}_i, \ \forall i, \forall i' \neq i \tag{5.6}$$

Table 5.1: List of Parameters of the CLP Formulation.

Symbol	Description
B	Number of bins
$\mathcal{B}$	Set of bins
$\mathrm{W}_{\max}$	Maximum weight supportable by TUs
L W H	Length, width, and height of TUs
$\mathrm{V}_{\max}$	Volume capacity of TUs
l_i w_i h_i	Length, width, and height of bin i
v_i	Volume of bin i
g_i	Weight of bin i
φ_i	Fragility index of bin i
σ_i	Priority index of bin i
γ	Maximum area gap between two stacked adjacent bins i and i'
δ	Maximum height difference between two laterally adjacent bins i and i'
λ_1, λ_2	Load balancing coefficients along the TU trasversal axes
ω_1, ω_2	Load balancing coefficients along the TU longitudinal axes

$$cz_{i'}p_{i'} - z_i p_i + \mathrm{H}\,\xi^u_{ii'} \leq \mathrm{H} - \mathrm{h}_{i'},\ \forall i, \forall i' \neq i. \tag{5.7}$$

2) TU dimension bounding constraints: the coordinates of bins must not exceed the dimensions of the TU:

$$c0 \leq x_i\, p_i \leq \mathrm{W} - \mathrm{w}_i,\ \forall i \tag{5.8}$$

$$c0 \leq y_i\, p_i \leq \mathrm{L} - \mathrm{l}_i,\ \forall i \tag{5.9}$$

$$c0 \leq z_i\, p_i \leq \mathrm{H} - \mathrm{h}_i,\ \forall i. \tag{5.10}$$

3) Bins relative positions constraints: a generic bin i can assume only one relative position with respect to a generic bin i' in the TU, that is: to the left, right, front, back, under or over:

$$c\xi^l_{ii'} + \xi^r_{ii'} + \xi^f_{ii'} + \xi^b_{ii'} + \xi^u_{ii'} + \xi^o_{ii'} = p_i\, p_{i'}, \forall i, \forall i' \neq i. \tag{5.11}$$

4) Bin rotation constraints: bins can rotate 90° in the horizontal plane:

$$\mathrm{w}'_i = \varrho_i\, \mathrm{l}_i + (1 - \varrho_i)\, \mathrm{w}_i,\ \forall i \tag{5.12}$$

$$\mathrm{l}'_i = \varrho_i\, \mathrm{w}_i + (1 - \varrho_i)\, \mathrm{l}_i,\ \forall i. \tag{5.13}$$

5) Bins stackability constraints: each bin i is associated a fragility index φ_i. Bins must then be stacked respecting an ascending fragility order, from the lowest to the highest one:

$$(\xi^o_{ii'} - \xi^u_{ii'})\, \varphi_{i'} \ \leq\ (\xi^o_{ii'} - \xi^u_{ii'})\, \varphi_i\,,\ \forall i, \forall i' \neq i. \tag{5.14}$$

6) Bin priority constraints: each bin i is assigned a priority index σ_i, which represents the priority of the delivery to which the bin belongs. The lowest the value of the priority index, the

Table 5.2: List of Decision Variables of the CLP Formulation.

Symbol	Description	Value
i, i'	Indices of bins	$\mathcal{B}$
p_i	Binary variable indicating if bin i is inside (1) or outside (0) TU	{0,1}
x_i	Coordinates of the lower right corner of the bin i	$[0, W]$
y_i		$[0, L]$
z_i		$[0, H]$
w'_i (l'_i)	X-axis (y-axis) dimension of bin i in the TU considering eventual vertical (horizontal) rotation	$[0, W]$ $[0, L]$
ϱ_i	Binary variable indicating if bin i is rotated (1) or not (0) with respect to vertical axis	{0,1}
$\xi^l_{i,i'}$ $\xi^r_{i,i'}$ $\xi^f_{i,i'}$ $\xi^b_{i,i'}$ $\xi^o_{i,i'}$ $\xi^u_{i,i'}$	Binary variables indicating if bin i is on the left of, on right of, in front of, behind, above, or below bin i' (1) or not (0)	{0,1}
X_i	Auxiliary variables introduced to linearize terms $x_i\, p_i$, $y_i\, p_i$, $z_i\, p_i$, $p_i\, p_{i'}$	$[0, W]$
Y_i		$[0, L]$
Z_i		$[0, H]$
$P_{ii'}$		{0,1}

highest the priority. It is considered here a LIFO positioning of bins inside the TU, thus bins are positioned in the TU starting from the rear up to the frontal part of the TU following a decreasing order of the priority index:

$$(\xi^b_{ii'} - \xi^f_{ii'})\, \sigma_{i'} \;\leq\; (\xi^b_{ii'} - \xi^f_{ii'})\, \sigma_i \;\; \forall i, \forall i' \neq i. \tag{5.15}$$

7) Bin horizontal stability constraints: the difference between the basis areas of two stacked and adjacent bins must be lower than or equal to a given threshold γ:

$$(\mathrm{l}_i \mathrm{w}_i - \mathrm{l}_{i'}\, \mathrm{w}_{i'})\, (\xi^o_{ii'} - \xi^u_{ii'}) \leq \gamma, \;\; \forall i, \forall i' \neq i. \tag{5.16}$$

8) Bin vertical stability constraints: the height difference between consecutive and adjacent bins must be lower than or equal to a given threshold δ:

$$(\mathrm{h}_i - \mathrm{h}_{i'})\, (\xi^f_{ii'} - \xi^b_{ii'}) \leq \delta, \;\; \forall i, \forall i' \neq i \tag{5.17}$$

$$-(\mathrm{h}_i - \mathrm{h}_{i'})\, (\xi^f_{ii'} - \xi^b_{ii'}) \leq \delta, \;\; \forall i, \forall i' \neq i. \tag{5.18}$$

9) TU volume capacity constraints: the total volume of bins loaded in the TU must not exceed its maximum volume:

$$\sum_{i \in \mathcal{B}} p_i \mathrm{v}_i \leq \mathrm{V}_{\max}. \tag{5.19}$$

10) TU weight capacity constraints: the total weight of bins loaded in the TU must not exceed its maximum admissible load:

$$\sum_{i \in \mathcal{B}} p_i \mathrm{g}_i \leq \mathrm{W}_{\max}. \tag{5.20}$$

11) Load balancing constraints: the load is positioned inside the TU so that its center of mass lies in a bounded area inside the basis of the TU, whose dimensions are related to the length and width of the TU by means of the λ_1 and λ_2 parameters for the transversal axes and by means of the ω_1 and ω_2 parameters for the longitudinal axes:

$$c\lambda_1 \mathrm{W} \leq \frac{\sum_i \mathrm{g}_i(x_i p_i + \mathrm{w}_i/2)}{\sum_i p_i \mathrm{g}_i} \leq \lambda_2 \mathrm{W}, \ \forall i \tag{5.21}$$

$$c\omega_1 \mathrm{L} \leq \frac{\sum_i \mathrm{g}_i(y_i p_i + \mathrm{l}_i/2)}{\sum_i p_i \mathrm{g}_i} \leq \omega_2 \mathrm{L}, \ \forall i. \tag{5.22}$$

In addition, the following integrality constraints hold:

$$cp_i \in \{0,1\}, \forall i \tag{5.23}$$

$$c\xi^l_{ii'}, \xi^r_{ii'}, \xi^f_{ii'}, \xi^b_{ii'}, \xi^u_{ii'}, \xi^o_{ii'} \in \{0,1\}, \forall i, \forall i' \neq i \tag{5.24}$$

$$c\varrho_i \in \{0,1\}, \forall i. \tag{5.25}$$

The resulting optimization problem (5.1)-(5.25) is a MINLP problem, due to the presence of the cross-products $x_i\, p_i$, $y_i\, p_i$, $z_i\, p_i$, $p_i\, p_{i'}$ in the constraints (5.2)-(5.11) and (5.21), (5.22). Using the approach proposed by Bemporad *et al.* in [23], a MILP problem can be obtained. In particular, it is introduced the auxiliary variables $X_i = x_i\, p_i$, $Y_i = y_i\, p_i$, $Z_i = z_i\, p_i$, $P_{ii'} = p_i\, p_{i'}$ and the corresponding following logical constraints:

$$X_i \leq (W - \mathrm{w}_i)\, p_i, \ \forall i \tag{5.26}$$

$$X_i \geq x_i - (W - \mathrm{w}_i)\,(1 - p_i), \ \forall i \tag{5.27}$$

$$Y_i \leq (L - \mathrm{l}_i)\, p_i, \ \forall i \tag{5.28}$$

$$Y_i \geq y_i - (L - \mathrm{l}_i)\,(1 - p_i), \ \forall i \tag{5.29}$$

$$Z_i \leq (H - \mathrm{h}_i)\, p_i, \ \forall i \tag{5.30}$$

$$Z_i \geq z_i - (H - \mathrm{h}_i)\,(1 - p_i), \ \forall i \tag{5.31}$$

$$-p_i + P_{ii'} \leq 0, \ \forall i, \forall i' \neq i \tag{5.32}$$

$$-p_{i'} + P_{ii'} \leq 0, \ \forall i, \forall i' \neq i \tag{5.33}$$

$$p_i + p_{i'} - P_{ii'} \leq 1, \ \forall i, \forall i' \neq i \tag{5.34}$$

Replacing X_i, Y_i, Z_i, $P_{ii'}$, in (5.2)-(5.11), and (5.21), (5.22) the mathematical problem (5.1)-(5.25) can be rewritten as follows:

$$\min \left(\mathrm{V}_{\max} - \sum_{i \in \mathcal{B}} p_i \, \mathrm{V}_i \right)$$

s.t. (5.12)-(5.20), (5.23)-(5.34)

$$X_i - X_{i'} + \mathrm{W}\, \xi^l_{ii'} \leq \mathrm{W} - \mathrm{w}_i, \ \forall i, \forall i' \neq i \tag{5.35}$$

$$X_{i'} - X_i + \mathrm{W}\, \xi^r_{ii'} \leq \mathrm{W} - \mathrm{w}_{i'}, \ \forall i, \forall i' \neq i \tag{5.36}$$

$$Y_i - Y_{i'} + \mathrm{L}\, \xi^b_{ii'} \leq \mathrm{L} - \mathrm{l}_i, \ \forall i, \forall i' \neq i \tag{5.37}$$

$$Y_{i'} - Y_i + \mathrm{L}\,\xi^f_{ii'} \leq \mathrm{L} - \mathrm{l}_{i'},\ \forall i, \forall i' \neq i \tag{5.38}$$

$$Z_i - Z_{i'} + \mathrm{H}\,\xi^o_{ii'} \leq \mathrm{H} - \mathrm{h}_i,\ \forall i, \forall i' \neq i \tag{5.39}$$

$$Z_{i'} - Z_i + \mathrm{H}\,\xi^u_{ii'} \leq \mathrm{H} - \mathrm{h}_{i'},\ \forall i, \forall i' \neq i \tag{5.40}$$

$$0 \leq X_i \leq \mathrm{W} - \mathrm{w}_i,\ \forall i \tag{5.41}$$

$$0 \leq Y_i \leq \mathrm{L} - \mathrm{l}_i,\ \forall i \tag{5.42}$$

$$0 \leq Z_i \leq \mathrm{H} - \mathrm{h}_i,\ \forall i \tag{5.43}$$

$$\xi^l_{ii'} + \xi^r_{ii'} + \xi^f_{ii'} + \xi^i_{ii'} + \xi^u_{ii'} + \xi^o_{ii} = P_{ii'}, \forall i, \forall i' \neq i \tag{5.44}$$

$$\lambda_1 \mathrm{W} \leq \frac{\sum_i \mathrm{g}_i (X_i + \mathrm{w}_i/2)}{\sum_i p_i \mathrm{g}_i} \leq \lambda_2 \mathrm{W},\ \forall i \tag{5.45}$$

$$\omega_1 \mathrm{L} \leq \frac{\sum_i \mathrm{g}_i (Y_i + \mathrm{l}_i/2)}{\sum_i p_i \mathrm{g}_i} \leq \omega_2 \mathrm{L},\ \forall i \tag{5.46}$$

The resulting MILP problem (5.35)-(5.46) consists in determining the 8B real and 7B(B-1)+2B binary variables, reported in Table 5.2, which minimize the objective function in (5.35) and meet the 5B bounding constraints (5.8)-(5.43), (5.45), (5.46), the 3B equality constraints (5.12), (5.13), (5.44), and the 25B inequality constraints (5.14)-(5.20), (5.23)-(5.34), (5.35)-(5.40).

5.3 The Proposed Algorithm for the Multi-drop Multi-CLP

This section describes the delivery-based algorithm that efficiently determines the optimal composition of TUs. The parameters, inputs, and outputs of the algorithm are shown in Table 5.3 respectively in the first, second and third section of the table. Specifically, a set of D deliveries are taken into account, whilst unlimited TUs are considered. The resolution procedure is shown in the flow chart of Fig. 5.1. In particular, the algorithm is based on a nested double-loop of iterations, which is indicated by *Loop 1* and *Loop 2* in Fig. 5.1.

Table 5.3: List of parameters, inputs and ouputs of the Multi-Drop Multi-CLP Algorithm.

Name	Description
Parameters	
$\mathrm{W}_{\max}$	Maximum weight supportable by TUs
$\mathrm{V}_{\max}$	Volume capacity of TUs
Inputs	
D	Number of deliveries ($D \geq 1$)
d	Index of deliveries ($d \in \{1, \ldots, D\}$)
$\mathcal{A}_d$	Set of bins associated to delivery $d \in \{1, \ldots, D\}$
i	Index of bins
v_i	Volume of bin i
g_i	Weight of bin i
Outputs	
T	Number of TUs to be loaded ($T \geq 1$)
t	Index of TUs ($t \in \{1, \ldots, T\}$)
$\mathcal{S}_t$	Set of bins to be loaded in TU $t \in \{1, \ldots, T\}$

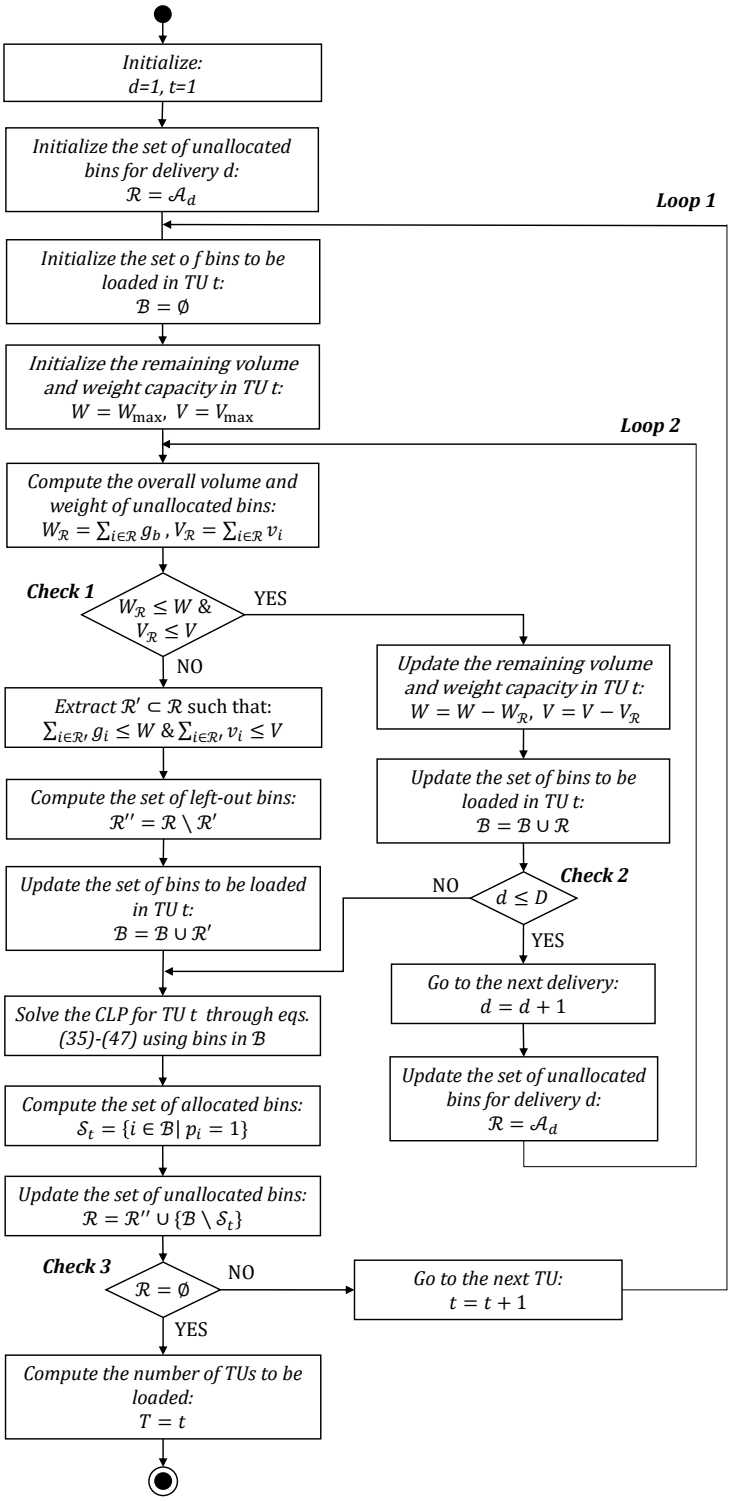


Figure 5.1: Flow chart of the proposed algorithm.

5.3.1 Iterating over TUs

The outer loop (i.e., *Loop 1*) in Fig. 5.1 of the algorithm iterates over the TUs and is composed by two phases. Referring to the t-th outer iteration, the first step aims at selecting the bins to be compatibly loaded in TU t; the latter aims at determining the optimal configuration for TU t by solving the single-CLP (5.35)-(5.46). These two steps are repeated until none of the given bins remains unallocated, as indicated by the decision block *Check 3* in Fig. 5.1.

5.3.2 Grouping/Ungrouping Bins

In this phase the given deliveries are inspected one at a time in order to select the set $\mathcal{B}$ of bins that could be loaded in TU t. Two cases may occur, as indicated by the decision block *Check 1* in Fig. 5.1. On the one hand, if the size of delivery d in terms of total weight and volume of the corresponding bins is higher than the available capacity of containers, only a portion of the delivery is assigned for the CLP of TU t. The remaining bins are thus deferred to the subsequent TU $t+1$. On the other hand, if the size of delivery d is lower than the available capacity of TUs, the inner loop (i.e., *Loop 2*) over the subsequent deliveries is performed to eventually group bins from various deliveries. This loop ends if no further deliveries remain, as indicated by the decision block *Check 1* and *Check 2* in Fig. 5.1, respectively.

5.3.3 Solving the CLP

Having selected a set $\mathcal{B}$ of bins that can be compatibly loaded in TU t, the optimal configuration for TU t is determined by solving the multi-drop single-CLP (5.35)-(5.46) defined in Section 5.3. In the case any of the selected bins are not allocated in the TU t, they are deferred to the subsequent TU $t+1$, as indicated by the decision block *Check 3* in Fig. 5.1.

5.4 Experimental Results

In this section the effectiveness of the proposed algorithm is tested on three realistic scenarios, for which it is assumed that the bins are packed over EU standard pallets, while the TUs can have three different ISO standards sizes. In the three scenarios, the number of considered shipments varies as well as the number of related bins and the corresponding dimensions (i.e., height and weight), so as to validate the obtained results and to analyze the computational effort of the proposed algorithm. In particular, with the aim of assessing the performance of the algorithm, the following performance indices are considered:

- $\mathrm{T_c}$: total computation time of the algorithm, i.e., the running time required to determine the number of TUs to be used and their corresponding composition in terms of identification and location of bins to be loaded;
- F_t: percentage fill ratio of the transport unit t, defined as $\mathrm{F}_t = \dfrac{\sum_{i\in\mathcal{B}} \mathrm{v}_i}{\mathrm{V}_{\max}} 100$, which represents the percentage of volume occupied by the bins in each TU t.

Note that the resolution of the multi-drop multi-CLP is implemented in the Matlab environment R2020b, on a macOS Big Sur PC with 2.3 GHz Intel Core i9 processor and 16GB RAM, and the optimization problem is solved using the Optimization Toolbox integrated in Matlab.

5.4.1 Scenarios Set-up

Hereafter it is reported the set-up of the three test scenarios specifying the input parameters to be provided for the execution of the proposed algorithm. It is assumed that bins are all packed on EU standard pallets sized $w_i = 1,200$ mm and $l_i = 800$ mm and have a maximal admissible height equal to $h_{max} = 2,400$ mm. Differently, the TUs have three possible ISO standard dimensions, namely, TU1 with $L_1 = 10,890$ mm $W_1 = 1,800$ mm $H_1 = 2,500$ mm; TU2 with $L_2 = 11,900$ mm $W_2 = 1,800$ mm $H_2 = 2,500$ mm, and TU3 with $L_3 = 12,100$ mm $W_3 = 1,800$ mm $H_3 = 2,500$ mm; the corresponding admissible weight is equal to $W_{max1} = 22,500$ kg, $W_{max2} = 40,000$ kg, and $W_{max3} = 42,500$ kg, respectively.

- *Scenario 1* - The first scenario considers a number of deliveries $D = 2$ composed respectively by 17 bins for the first one and 27 bins for the second one; the height of bins varies in the set $h_i = \{h_{max}, h_{max}/2, h_{max}/3\}$ where h_{max} is equal to 2,400 mm; the available TUs are TU1 and TU2. Note that there are:
  - 13 bins with $h_i = h_{max}$ and $w_i = 1500$ kg;
  - 20 bins with $h_i = h_{max}/2$ and $w_i = 750$ kg;
  - 11 bins with $h_i = h_{max}/3$ and $w_i = 325$ kg.
- *Scenario 2* - The second scenario considers a number of deliveries $D = 3$ composed respectively by 17 bins for the first one, 23 bins for the second one, and 28 for the third one, whereas the height of bins varies in the set $h_i = \{h_{max}, h_{max}/2, h_{max}/3\}$ where h_{max} is equal to 2,400 mm; the available TUs are TU1, TU2 and TU3. Note that there are:
  - 13 bins with $h_i = h_{max}$ and $w_i = 1500$ kg;
  - 39 bins with $h_i = h_{max}/2$ and $w_i = 750$ kg;
  - 16 bins with $h_i = h_{max}/3$ and $w_i = 325$ kg.
- *Scenario 3* - The third scenario considers a number of deliveries $D = 7$ composed respectively by 17 bins for the first one, 25 bins for the second one, 29 for the third one, 6 for the fourth one, 10 for the fifth one, 6 for the sixth one and 7 for the seventh one, whereas the height of the bins varies in the set $h_i = \{h_{max}, h_{max}/2, h_{max}/3\}$ where h_{max} is equal to 2,400 mm; the available TUs are TU1, TU2, TU3 and TU4. Note that there are:
  - 20 bins with $h_i = h_{max}$ and $w_i = 1500$ kg;
  - 52 bins with $h_i = h_{max}/2$; and $w_i = 750$ kg;
  - 28 bins with $h_i = h_{max}/3$ and $w_i = 325$ kg.

For all the considered scenarios, the parameters referring to the stability and the load balancing constraints get the following values: $\delta = h_{max}/2$, $\gamma = 1920$, $\lambda_1 = \omega_1 = 1/4$, $\lambda_2 = \omega_2 = 3/4$.

5.4.2 Results Analysis and Discussion

In this section, the results obtained in the considered scenarios are discussed and summarized in Table 5.4. In particular, for each scenario it is reported the computation time required by the algorithm, i.e., T_c; the TU index t; the TU type, namely, TU1, TU2, or TU3; the fill ratio in percentage for each loaded transport unit t, i.e., F_t; the set of delivery indices associated to the bins loaded in each TU.

When comparing Scenario 1 and Scenario 2, it emerges that increasing of 40% the load total volume and the number of the deliveries implies an increase of about 60% of the computation

Table 5.4: Comparison of the considered scenarios

Scenario	T_c [s]	TU index t	TU type	F_t (%)	Set of deliveries indices d
Scenario 1	152.5	1	TU1	70.9	1
		2	TU2	64.6	2
Scenario 2	243.7	1	TU1	69.8	1
		2	TU2	55.1	2
		3	TU3	60.3	2, 3
Scenario 3	521.3	1	TU1	70.5	1
		2	TU2	55.3	1, 2
		3	TU3	60.5	2, 3
		4	TU1	70.9	4, 5, 6, 7

time. In addition, it is possible to notice that in Scenario 1 the bins of $d = 1$ are all loaded in the TU $t = 1$; consequently in the second TU $t = 2$ only bins of the second delivery $d = 2$ are present. Differently, in Scenario 2 the bins of delivery $d = 2$ are partitioned over two different transport units. This highlights the suitability of the proposed algorithm for the multi-drop problem as well as the fulfillment of the multi-drop constraints. As for the comparison between Scenario 1 and Scenario 3, it emerges that increasing of almost 100% the load total volume and of 130% the number of the deliveries implies an increase of about 240% of the computation time. Moreover, in Scenario 3 the bins of $d = 1$ and of $d = 2$ are partitioned over two different transport units, demonstrating again the fulfillment of multi-drop. Finally, for the sake of showing the fullfillment of the bin priority constraints, in Fig. 5.2 it is reported the detailed output of Scenario 3. In particular, in Fig. 5.2.a it is represented the transport unit of type TU1 loaded with the bins belonging to delivery $d = 1$. The remaining bins are loaded in TU2 reported in Fig. 5.2.b, where they are grouped and combined with the delivery $d = 2$. Similarly, the remaining bins of delivery $d = 2$ are loaded in a further transport unit type TU3, where they are grouped and combined with the delivery $d = 3$, as shown in Fig. 5.2.c. Finally, deliveries $d = 4$, 5, 6 are grouped and positioned in the fourth transport unit type TU1, as reported in Fig. 5.2.

5.5 Conclusion

This chapter investigates the multi-drop multi-container loading problem (CLP), i.e., the problem of optimally packing multiple bins associated to multiple deliveries into a finite number of transport units (TUs). A novel mixed integer linear programming delivery-based algorithm is proposed to efficiently determine the optimal composition of TUs, while minimizing the unused space, fulfilling a set of geometric and safety constraints, and complying with the delivery allocation. Several realistic case studies show the effectiveness of the presented approach in determining stable and compact TU loading configurations in a short computation time.

Future research will address enhancing the algorithm performance by introducing heuristic or meta-heuristic optimization solutions and integrating the CLP with the vehicle routing problem.

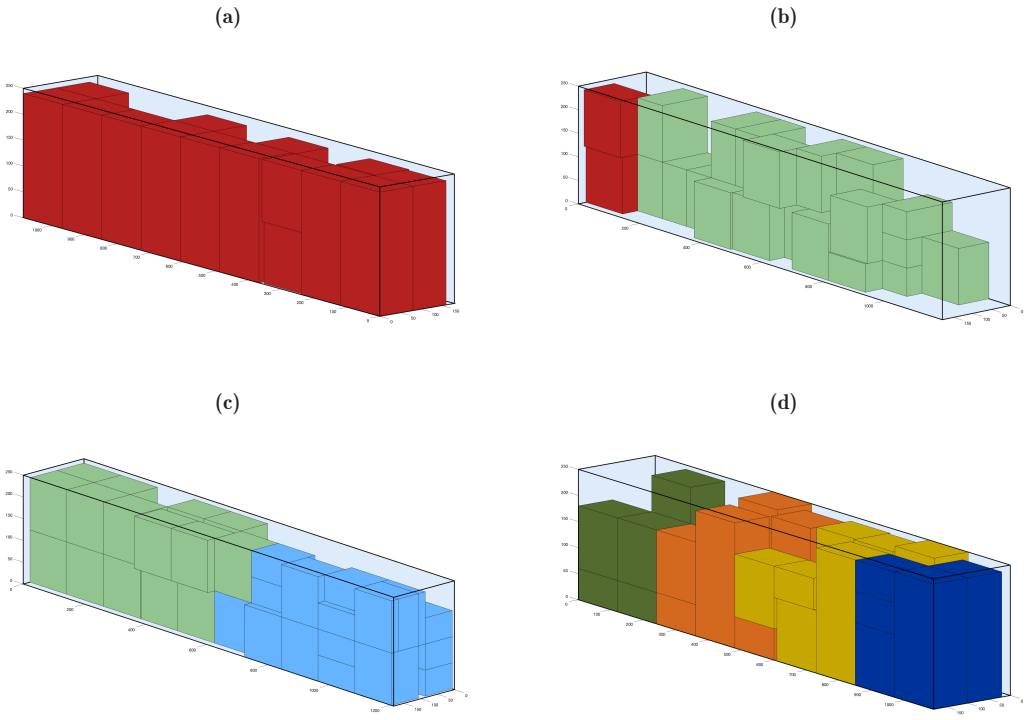


Figure 5.2: TUs loading in Scenario 3: bins composition in the first TU (a), second TU (b), third TU (c), and fourth TU (d).

References

[1] Strandhagen, J. O., Vallandingham, L. R., Fragapane, G., Strandhagen, J. W., Stangeland, A. B. H., and Sharma, N., "Logistics 4.0 and emerging sustainable business models," *Advances in Manufacturing*, vol. 5, no. 4, pp. 359–369, 2017.

[2] Digiesi, S., Facchini, F., Mossa, G., and Mummolo, G., "Minimizing and balancing ergonomic risk of workers of an assembly line by job rotation: A minlp model," *International Journal of Industrial Engineering and Management*, vol. 9, no. 3, pp. 129–138, 2018.

[3] Bischoff, E. E. and Ratcliff, M., "Issues in the development of approaches to container loading," *Omega*, vol. 23, no. 4, pp. 377–390, 1995.

[4] Facchini, F., Digiesi, S., and Mossa, G., "Optimal dry port configuration for container terminals: A non-linear model for sustainable decision making," *International Journal of Production Economics*, vol. 219, pp. 164–178, 2020.

[5] Wäscher, G., Haußner, H., and Schumann, H., "An improved typology of cutting and packing problems," *European journal of operational research*, vol. 183, no. 3, pp. 1109–1130, 2007.

[6] Bortfeldt, A. and Wäscher, G., "Constraints in container loading–a state-of-the-art review," *European Journal of Operational Research*, vol. 229, no. 1, pp. 1–20, 2013.

[7] Scheithauer, G., Terno, J., Riehme, J., and Sommerweiss, U., "A new heuristic approach for solving the multi-pallet packing problem," *Dresden: Technische Universität Dresden*, 1996.

[8] Pisinger, D., "Heuristics for the container loading problem," *European journal of operational research*, vol. 141, no. 2, pp. 382–392, 2002.

[9] Moura, A. and Oliveira, J. F., "A grasp approach to the container-loading problem," *IEEE Intelligent Systems*, vol. 20, no. 4, pp. 50–57, 2005.

[10] Jin, Z., Ohno, K., and Du, J., "An efficient approach for the three-dimensional container packing problem with practical constraints," *Asia-Pacific Journal of Operational Research*, vol. 21, no. 03, pp. 279–295, 2004.

[11] Lin, J.-L., Chang, C.-H., and Yang, J.-Y., "A study of optimal system for multiple-constraint multiple-container packing problems," in *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*, Springer, 2006, pp. 1200–1210.

[12] Ceschia, S. and Schaerf, A., "Local search for a multi-drop multi-container loading problem," *Journal of Heuristics*, vol. 19, no. 2, pp. 275–294, 2013.

[13] Alonso, M., Alvarez-Valdes, R., Iori, M., and Parreño, F., "Mathematical models for multi container loading problems with practical constraints," *Computers & Industrial Engineering*, vol. 127, pp. 722–733, 2019.

[14] Nascimento, O. X. do, Queiroz, T. A. de, and Junqueira, L., "Practical constraints in the container loading problem: Comprehensive formulations and exact algorithm," *Computers & Operations Research*, vol. 128, p. 105 186, 2021.

[15] Dotoli, M., Epicoco, N., Falagario, M., Seatzu, C., and Turchiano, B., "A decision support system for optimizing operations at intermodal railroad terminals," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 3, pp. 487–501, 2016.

[16] Dotoli, M. and Epicoco, N., "A technique for the optimal management of containers' drayage at intermodal terminals," in *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, IEEE, 2016, pp. 000 566–000 571.

[17] Junqueira, L., Morabito, R., and Yamashita, D. S., "Mip-based approaches for the container loading problem with multi-drop constraints," *Annals of Operations Research*, vol. 199, no. 1, pp. 51–75, 2012.

[18] Lai, K., Xue, J., and Xu, B., "Container packing in a multi-customer delivering operation," *Computers & industrial engineering*, vol. 35, no. 1-2, pp. 323–326, 1998.

[19] Christensen, S. G. and Rousøe, D. M., "Container loading with multi-drop constraints," *International Transactions in Operational Research*, vol. 16, no. 6, pp. 727–743, 2009.

[20] Gendreau, M., Iori, M., Laporte, G., and Martello, S., "A tabu search algorithm for a routing and container loading problem," *Transportation Science*, vol. 40, no. 3, pp. 342–350, 2006.

[21] Tarantilis, C. D., Zachariadis, E. E., and Kiranoudis, C. T., "A hybrid metaheuristic algorithm for the integrated vehicle routing and three-dimensional container-loading problem," *IEEE Transactions on Intelligent Transportation Systems*, vol. 10, no. 2, pp. 255–271, 2009.

[22] Fuellerer, G., Doerner, K. F., Hartl, R. F., and Iori, M., "Metaheuristics for vehicle routing problems with three-dimensional loading constraints," *European Journal of Operational Research*, vol. 201, no. 3, pp. 751–759, 2010.

[23] Bemporad, A. and Morari, M., "Control of systems integrating logic, dynamics, and constraints," *Automatica*, vol. 35, no. 3, pp. 407–427, 1999.

Chapter 6

Logistics 4.0: A Matheuristics for the Integrated Vehicle Routing and Container Loading Problem

Abstract

The increasing demand for freight transport requires logistic companies to improve their competitiveness by ensuring high service levels at limited costs. This chapter investigates the problem of defining delivery plans with the aim of supporting logistic companies in reducing planning times and freight delivery costs. In delivery planning, given a set of delivery requests, both the routes and load configurations of Transport Units (TUs) are to be established. In the literature, this problem is defined as Three-dimensional Loading Capacitated Vehicle Routing Problem with Time Windows (3L-CVRPTW) and it is the integration of two different well-known literature problems: the vehicle routing problem and the container loading problem that are generally tackled separately. Moreover, only a few contributions present solution approaches for real logistic systems, and these methods are mainly based on heuristics. In this chapter, starting with the CLP algorithm described in Chapter 5, a matheuristic algorithm for the integrated solution of the vehicle routing problem and container loading problem is defined. The proposed method is suitable for real logistic applications and combines the advantages of exact solutions with the rapidity of heuristics. The approach aims at minimizing the total travel costs and the clients' time windows violations in the routes' definition while optimizing the configuration of the cargo inside each TU. The developed matheuristic algorithm is tested both on a well-known literature benchmark and on a real dataset provided by the Italian company E80 Group. The obtained results show that the proposed method succeeds in determining in a short computational time both feasible routes and loading plans, minimizing the related costs while fulfilling logistics constraints.

Contents

6.1 Introduction

Logistics 4.0 is a highly competitive branch of Industry 4.0 that involves a wide variety of sectors, including retail sales, shipments, waste management, and construction. Its main aim is to improve productivity and the quality of service by applying the enabling technologies of Industry 4.0 (e.g., robotics, automation, Internet of Things) to logistics and supply chains [1] from the storing of

objects in warehouses, to their loading in Transport Units (TUs) and its subsequent delivery to the clients. This chapter is focused on distribution logistics which generates the highest percentage of the logistic operations costs. In particular, it addresses the complex problem of jointly planning delivery routes and loading plans for Transport Units (TUs) of logistic companies. In the related literature the two problems are generally tackled separately as the Vehicle Routing Problem (VRP) and the Container Loading Problem (CLP) (e.g., in [2] [3]). The former, given a set of clients to be visited by a set of TUs (and TUs types), aims at finding the routes that optimize a proper objective function (e.g., the traveled distance, the consumption of fuel), while satisfying routing constraints. The latter belongs to the wider class of cutting and packing problems, whose aim is to optimize the occupation of containers ensuring that boxes do not overlap and lie entirely inside the container. In this case, the aim is to reduce the total cost of delivery, which entails fixed costs (mainly driver wages and/or renting fees) and variable costs (directly dependent on the distance traveled such as the consumption of fuel) combined with particular attention to the quality of service, which basically ensures that all addresses must be served in predetermined time windows which correspond to the opening hours of clients' warehouses.

The seminal version of the VRP was the "truck dispatching problem", formulated in [4], later named "Vehicle Routing Problem" in [5], whose objective is to find the minimal travel cost to serve a set of clients and their associated demand, so that each client is visited and served by a TU only once. A comprehensive survey on this topic is presented in [6], which classifies the VRP formulations according to different aspects, such as the number of stops/clients, the possibility to include time window restrictions and travel time, the used data and the type of information characteristics (i.e., evolution, quality, and availability of information). A common extension of the basic VRP is the Capacitated Vehicle Routing Problem with Time Windows (CVRPTW) which includes capacity and time windows constraints, respectively for the TUs and the delivery to clients. However, this model might result in a too simplistic representation of the routing problem [7], [8], [9]. As for the CLP, relevant contributions are proposed in [10] and [11]: the former presents a reference typology for the cutting and packing problems, which classifies the different problems according to the type of objective function, as well as the heterogeneity, number, and dimensions of both the boxes and the containers, the latter examines the state of the art on CLP and classifies the different types of constraints developed in the last years with particular attention to the ones used in real applications. The combination of CVRPTW with the CLP in the three-dimensional space leads to the Three-Dimensional Loading Capacitated Vehicle Routing Problem with Time Windows (3L-CVRPTW).

In the same direction, this chapter proposes a novel matheuristic approach for the 3L-CVRPTW. The method consists of a heuristic combined with the exact solution of three Mixed Integer Linear Programming (MILP) problems that take into account logistic constraints. The first problem regards the selection of the proper number and type of TUs to be assigned to clients; the second aims at solving the CVRPTW by minimizing the travel costs and time windows violations; and the third, given the set of clients assigned to a TU resulting from the CVRPTW problem solution, aims at optimizing the loading plan of the TU. Compared to the above-mentioned works, it is provided a more comprehensive approach applicable to the logistic sector that not only considers the traditional VRP and CLP constraints, but also additional logistic loading requirements, i.e, the selection of the proper TUs to be used for the delivery, the balancing of the cargo, the static and dynamic stability, and the management of the LIFO (Last In- First Out) filling. Moreover, the proposed method allows to jointly tackle the VRP and CLP problems in a short computational time, thanks to the combination of mathematical programming and heuristics, thus overcoming the limitations that emerged from the literature. On the one hand, this allows avoiding simplifying assumptions for the mathematical formalization of the 3L-CVRPTW, which would lead to the representation of only a limited set of logistic rules. On the other hand, this allows avoiding to

address the problem only with heuristic approaches, which would disregard the mathematical programming model of the problem requiring the definition of ad hoc solutions. The remainder of this chapter is organised as follows. In Section 6.2 the MILP formulations of the CVRPTW and CLP is presented, while in Section 6.3 the proposed matheuristics is describe. Section 6.4 presents the experimental tests and discusses the obtained results. Finally, Section 6.5 concludes the Chapter.

6.2 Mathematical Formulation of VRP and CLP.

This section presents the problem formulations used in the proposed matheuristics. The former problem, formulated as a CVRPTW, aims at finding the optimal route for each TU in terms of total travel costs and time windows violations while fulfilling a set of temporal and capacity constraints. The latter problem aims at finding the optimal configuration of the bins inside each TU minimizing the empty space while fulfilling geometrical and safety constraints. As it will be explained in the next sections, we consider the two problems separately, although the results obtained by the first one are used as input of the second one in the matheuristic approach (e.g. the priority S_b of each bin which is assigned according to the routing plan).

6.2.1 The Capacitated Vehicle Routing Problem with Time Windows formulation

The routing problem is formulated as a multiobjective optimization problem that aims at minimizing both the time and economical transport costs, while satisfying logistic constraints. Most of them are related to the feasibility of the routes, for example they ensure that each client must be visited only one time and by one TU, that each TU must start from the initial depot and end at the final depot or also avoid the creation of a loop inside the algorithm. Some of them are related to the capacity of the TUs both in terms of volume and total weight, while others define the maximum number of TUs. For the proposed formulation, the following assumptions hold: multiple size TUs are available, a single TU can contain the cargo of multiple clients, and the cargo of a client always fits a single TU. The limitations imposed by the last assumption are overcome by the proposed matheuristics, as will be explained in Section 6.3. The parameters and variables of the problem are respectively reported in Tables 6.1, 6.2 and 6.3. Note that, the objective function (6.1) is the linear combination of three fundamental costs associated to the logistic transport: the first term is the sum of the base costs associated to each TU assigned to a route, which can include rental fees, maintenance, etc., and are not related with the traveled distance; the second term is the sum of the variable costs associated to each TU, such as the gasoline consumption, which are related with the traveled distance; the third term represents a penalty cost assigned to the TUs in case of early arrival with respect to clients' time window. Note that the third term is particularly useful to avoid the waiting time of drivers for the opening of the client's warehouse, and thus to obtain a more efficient route.

The CVRPTW is formulated as follows:

$$\min_{k}(\mathrm{f}_1 \sum_{k=1}^{\mathrm{M}} \sum_{j=1}^{\mathrm{N}} \mathrm{C}^k t_{0j}^k + \mathrm{f}_2 \sum_{k=1}^{\mathrm{M}} \sum_{i=1}^{\mathrm{N}} \sum_{j=1}^{\mathrm{N}} \mathrm{V}^k \mathrm{D}_{i,j}\ t_{i,j}^k + \mathrm{f}_3 \sum_{k=1}^{\mathrm{M}} \sum_{j=1}^{\mathrm{N}} \delta_j^k) \tag{6.1}$$

$$\tag{6.2}$$

subject to:

$$\sum_{i=0}^{\mathrm{N}} t_{i,l}^k - \sum_{j=1}^{(\mathrm{N}+1)} t_{l,j}^k = 0,\ \forall k \in \mathcal{M}, l \in \mathcal{N},\ i \neq j \tag{6.3}$$

Table 6.1: List of common parameters of the CVRPTW and CLP formulations.

Name	Description
M	Number of TUs
N	Number of clients
B	Number of bins
ι_b	Length of bin b
χ_b	Width of bin b
γ_b	Height of bin b
$\mathrm{w}_{b,i}$	Weight of bin b belonging to client i
L_k	Length of TU k
W_k	Width of TU k
H_k	Height of TU k
Q_k	Maximum weight supported by TU k

Table 6.2: List of parameters of the CVRPTW formulation.

Name	Description
f_1	Weight of the base cost term of the obj. function
f_2	Weight of the variable cost term of the obj. function
f_3	Weight of the delivery delay term of the obj. function
V^k	Variable costs of TU k
C^k	Base costs of TU k
$\mathrm{D}_{i,j}$	Kilometers from client i to j
$\mathrm{E}_{i,j}$	Travel time from client i to j
F_i^k	Service time of TU k at client i
R	Maximum number of TUs to be used
A_i	Opening time of time window of client i
Z_i	Closing time of time window of client i
U	Upper bound for arrival time at each client

$$\sum_{k=1}^{\mathrm{M}} \sum_{i=1}^{\mathrm{N}} t_{i,l}^k = 1,\ \forall l \in n, l \neq i \tag{6.4}$$

$$\sum_{j=1}^{\mathrm{N}} t_{0,j}^k = 1,\ \forall k \in m \tag{6.5}$$

$$\sum_{i=1}^{\mathrm{N}} t_{i,(\mathrm{N}+1)}^k = 1,\ \forall k \in m \tag{6.6}$$

$$\sum_{j=1}^{\mathrm{N}} \sum_{i=1}^{\mathrm{N}} \sum_{b=1}^{\mathrm{B}} \mathrm{w}_{b,j}\ t_{i,j}^k \leq \mathrm{Q}_k, \forall k \in m \tag{6.7}$$

$$\sum_{j=1}^{\mathrm{N}} \sum_{i=1}^{\mathrm{N}} \sum_{b=1}^{\mathrm{B}} \chi_{b,j}\ \iota_{b,j}\ \gamma_{b,j}\ t_{i,j} \leq \mathrm{W}_k \mathrm{H}_k \mathrm{L}_k,\ \forall k \in m \tag{6.8}$$

Table 6.3: List of variables of the CVRPTW formulation.

Name	Description	Definition set
k	Indices of TUs	$\mathcal{M}$={1,2,...,M}
i, j, l	Indices of clients	$\mathcal{N}$={1,2,...,N}
b, b'	Indices of bins	$\mathcal{B}$={1,2,...,B}
$t^k_{i,j}$	The k-th TU runs from client i to client j (1) or not (0)	{0,1}
$t^k_{0,j}$	the k-th TU runs from the initial depot to client j (1)or not (0)	{0,1}
$t^k_{i,(\mathrm{N}+1)}$	the k-th TU runs from client i to the final depot (1) or not (0)	{0,1}
s^k_j	Arrival time of the k-th TU at client j	$\mathbb{R}^+$
δ^k_j	Advance/delay of the k-th TU arrival at client j with respect to opening time A_j	$\mathbb{R}^+$

$$\delta^k_i \geq \mathrm{A}_i - s^k_i, \ \forall k \in \mathcal{M}, \forall i \in \mathcal{N} \tag{6.9}$$

$$s^k_i \leq \mathrm{Z}_i - \mathrm{F}^k_i, \ \forall k \in \mathcal{M}, \forall i, j \in \mathcal{N} \tag{6.10}$$

$$s^k_i + \mathrm{F}^k_i + \mathrm{E}_{i,j}\ t^k_{i,j} - s^k_j + \mathrm{U}\ t^k_{i,j} \leq \mathrm{U}, \ \forall k \in \mathcal{M}, \forall i, j \in \mathcal{N} \tag{6.11}$$

$$s^k_i + \mathrm{F}_i + \mathrm{E}_{i,j}\ t^k_{i,(\mathrm{N}+1)} - \mathrm{Z}_{\mathrm{N}+1} + \mathrm{U}\ t^k_{i,j} \leq \mathrm{U}, \tag{6.12}$$
$$\forall k \in \mathcal{M}, \forall i, j \in \mathcal{N}$$

where $\mathrm{U} = max_{i,j}\ \{\mathrm{Z}_i - \mathrm{A}_i + \mathrm{E}_{i,j}\}$.

The three terms in (6.1) are weighted by parameters f_1, f_2, and f_3, whose sum is equal to 1 and whose value depends on the specific application. Constraints (6.3)-(6.6) are related to the feasibility of the routes: (6.3) impose that the sum of the routes entering a client l must be equal to the sum of the routes exiting the same client and prohibits both the creation of loops on the same client i and in the network (i.e., the TU k can only go from i to j or from j to i); (6.4) impose that each client must be visited only once. Constraints (6.5) and (6.6) are related to the starting and ending clients of each route and ensure that each TU must start its route from the initial depot (indicated by index 0) and end it at the final depot (indicated by index N+1). Then, constraints (6.7) and (6.8) are related to the capacity of the TUs: the former ensures that the total weight of the clients' bins does not exceed the maximum supported weight of the TU in which they are loaded; the latter ensures that the total volume of the bins of the clients does not exceed the volume of the TU in which they are loaded. The last constraints (6.9)-(6.12) are related to the time windows, which are defined as a pair of values (A_i, Z_i) where A_i is the opening time for delivery of client i and Z_i is the closing time. Constraints (6.9) impose that the penalty δ^k_i is greater than 0 if the TU k arrives before client's i opening time. Constraints (6.10) are hard constraints and impose that each TU must arrive at each related client before its closing time, while (6.11) and (6.12) impose that the arrival time of each TU k to each client j must be higher than or equal to the sum of the arrival time at client i, the service time at client i and the travel time from client i to client j.

6.2.2 The Container Loading Problem Formulation

The aim of the CLP problem is to find the optimal configuration of the bins inside each TU according to the results obtained for problem (6.1)-(6.12) and fulfilling a set of geometric and safety constraints. It is assumed that only TU is available and the considered bins belong to a single client and can exceed the capacity of the TU (in terms of volume and/or weight). Thus the optimization problem aims at computing the configuration of the bins that minimizes the empty space in the TU. The parameters and variables used for the formulation of this problem are reported respectively in Tables 6.4 and 6.5 (second subtable).

Table 6.4: List of parameters of the CLP formulation.

Name	Description
O	An arbitrary large number
F_b	Fragility of bin b
S_b	Priority of bin b
G	Maximum area gap between consecutive bins
H	Maximum height gap between adjacent bins
λ_1, λ_2	Load balancing along the TU transversal axes
ω_1, ω_2	Load balancing along the TU longitudinal axes

Table 6.5: List of variables of the CLP formulation.

Name	Description	Definition set
b, b'	Indices of bins	$\mathcal{B}$ ={1,2,...,B}
p_b	bin b is inside (1) or outside (0) TU	{0,1}
x_b	x-axis coordinate of the left-bottom corner of bin b	$\mathbb{R}^+$
y_b	y-axis coordinate of the left-bottom corner of bin b	$\mathbb{R}^+$
z_b	z-axis coordinate of the left-bottom corner of bin b	$\mathbb{R}^+$
l_{xb}	The length of bin b is parallel to the x-axis (1) or not (0)	{0,1}
l_{yb}	the length of bin b is parallel to the y-axis (1) or not (0)	{0,1}
$le_{b,b'}$	bin b is on the left (1) of bin b' or not (0)	{0,1}
$r_{b,b'}$	bin b is on the right (1) of bin b' or not (0)	{0,1}
$b_{b,b'}$	bin b is back (1) bin b' or not (0)	{0,1}
$f_{b,b'}$	bin b is in front of (1) bin b' or not (0)	{0,1}
$o_{b,b'}$	bin b is over (1) bin b' or not (0)	{0,1}
$u_{b,b'}$	bin b is under (1) bin b' or not (0)	{0,1}

The CLP is defined as follows:

$$\min(\mathrm{L}\ \mathrm{W}\ \mathrm{H} - \sum_{b=1}^{\mathrm{B}} p_b\ \iota_b\ \chi_b\ \gamma_b) \tag{6.13}$$

subject to:

$$le_{b,b'} + r_{b,b'} + f_{b,b'} + b_{b,b'} + u_{b,b'} + o_{b,b'} = p_b + p_{b'} - 1, \quad \forall b, b' \in \mathcal{B}, b' \leq b \tag{6.14}$$

$$x_b + \iota_b l_{\mathrm{x}b} + \chi_b(1 - l_{\mathrm{x}b}) \leq x_{b'} + (1 - le_{b,b'})\mathrm{O}, \quad \forall b, b' \in \mathcal{B}, b' \leq b \tag{6.15}$$

$$x_{b'} + \iota_{b'} l_{\mathrm{x}b'} + \chi_{b'}(1 - l_{\mathrm{x}b'}) \leq x_b + (1 - r_{b,b'})\mathrm{O}, \quad \forall b, b' \in \mathcal{B}, b' \leq b \tag{6.16}$$

$$y_b + \chi_b(1 - l_{\mathrm{y}b}) + \iota_b l_{\mathrm{y}b} \leq y_{b'} + (1 - b_{b,b'})\mathrm{O}, \quad \forall b, b' \in \mathcal{B}, b' \leq b \tag{6.17}$$

$$y_{b'} + \chi_{b'}(1 - l_{\mathrm{y}b'}) + \iota_{b'} l_{\mathrm{y}b'} \leq y_b + (1 - f_{b,b'})\mathrm{O}, \quad \forall b, b' \in \mathcal{B}, b' \leq b \tag{6.18}$$

$$z_b + \gamma_b \leq z_{b'} + (1 - o_{b,b'})\mathrm{O}, \quad \forall b, b' \in \mathcal{B}, b' \leq b \tag{6.19}$$

$$z_{b'} + \gamma_{b'} \leq z_b + (1 - u_{b,b'})\mathrm{O},\ \forall b, b' \in \mathcal{B}, b' \leq b \tag{6.20}$$

$$x_b + \iota_b l_{\mathrm{x}b} + \chi_b(1 - l_{\mathrm{x}b}) \leq \mathrm{W} + (1 - p_b)\mathrm{O},\ \forall b \in \mathcal{B} \tag{6.21}$$

$$y_b + \chi_b(1 - l_{\mathrm{y}b}) + \iota_b l_{\mathrm{y}b} \leq \mathrm{L} + (1 - p_b)\mathrm{O},\ \forall b \in \mathcal{B} \tag{6.22}$$

$$z_b + \gamma_b \leq H + (1 - p_b)\mathrm{O},\ \forall b \in \mathcal{B} \tag{6.23}$$

$$l_{\mathrm{x}b} + l_{\mathrm{y}b} = 1,\ \forall b \in \mathcal{B} \tag{6.24}$$

$$(o_{b,b'} - u_{b,b'})F_{b'} \leq (o_{b,b'} - u_{b,b'})F_b,\ \forall b, b' \in \mathcal{B}, b' \neq b \tag{6.25}$$

$$(b_{b,b'} - f_{b,b'})S_{b'} \leq (b_{b,b'} - f_{b,b'})S_b,\ \forall b, b' \in \mathcal{B}, b' \leq b \tag{6.26}$$

$$(\iota_b\chi_b - \iota_{b'}\chi_{b'})(o_{b,b'} - u_{b,b'})\mathrm{G},\ \forall b, b' \in \mathcal{B}, b' \leq b \tag{6.27}$$

$$(\gamma_b - \gamma_{b'})(f_{b,b'} - b_{b,b'}) \leq \mathrm{H},\ \forall b, b' \in \mathcal{B}, b' \leq b \tag{6.28}$$

$$-(\gamma_b - \gamma_{b'})(f_{b,b'} - b_{b,b'})\mathrm{H},\ \forall b, b' \in \mathcal{B}, b' \leq b \tag{6.29}$$

$$\sum_{b=1}^{\mathrm{B}} p_b\gamma_b\chi_b\iota_b \leq \mathrm{V} \tag{6.30}$$

$$\sum_{b=1}^{\mathrm{B}} p_b\mathrm{w}_b \leq \mathrm{Q} \tag{6.31}$$

$$\lambda_1\mathrm{W} \leq \frac{\sum_b \mathrm{w}_b(x_b + \chi_b/2)}{\sum_b p_b\mathrm{w}_b} \leq \lambda_2\mathrm{W},\ \forall b \in \mathcal{B} \tag{6.32}$$

$$\omega_1\mathrm{L} \leq \frac{\sum_b \mathrm{w}_b(y_b + \iota_b/2)}{\sum_b p_b\mathrm{w}_b} \leq \omega_2\mathrm{L},\ \forall b \in \mathcal{B}. \tag{6.33}$$

Note that the objective function (6.13) is the empty space inside the TU given the total volume of bins assigned to the TU. Constraints (6.14) impose that between two different bins, a bin can assume only one relative position with respect to the other one in the TU, that is: on the left, right, front, back, under, or over. Constraints (6.15) to (6.20) ensure that the coordinates of two bins do not overlap, while constraints (6.21) to (6.23) guarantee that the coordinates of bins

do not exceed the dimensions of the TU and let the bin rotate 90 degrees in order to optimise the space. Constraints (6.24) ensure the unique assignment of the orientation of each item i. On the other hand, constraints (6.25) to (6.29) determine the positioning of the bins inside the TU, in particular, (6.25) orders them by height according to their fragility index with an ascending order, (6.26) group and order them according to their priority index (i.e., the priority of the delivery to which the bin belongs so that the ones with the lowest values are the nearest to the door and so the first to be delivered), (6.27) guarantees horizontal stability (i.e., the difference between the basis areas of two stacked and adjacent bins must be lower than or equal to a given threshold) while (6.28) and (6.29) ensure that the items do not move while the TU is moving, by imposing that the height difference between consecutive and adjacent bins is lower than or equal to a given threshold. Constraints (6.30) and (6.31) are instead related to the capacity of the TU both in terms of total volume loaded (6.30) and maximum supported weight (6.31). In particular, it is imposed that the sum of the volume and the weight of all the bins of the different clients assigned to each TU, fulfill its capacity. Finally, (6.32) and (6.33) are load balancing constraints. In order to achieve load balancing, the load is positioned inside the TU so that its center of mass lies in a bounded area inside the basis of the TU, whose dimensions are related to the length and width of the TU by means of the λ_1 and λ_2 parameters for the transversal axes and by means of the ω_1 and ω_2 parameters for the longitudinal axes.

6.3 The Proposed Matheuristics for the 3L-CVRPTW Problem

The proposed method consists of a heuristics combined with the exact solution of the mathematical problems presented in Subsections 6.2.1 and 6.2.2, and of an additional problem aimed at the proper selection of TUs for clients with large cargoes. The purpose of the matheuristics is to obtain feasible routing and loading plans for a set of TUs that must satisfy the delivery requests of a set of clients. Instead of solving the delivery planning as a complex optimization problem including a large set of constraints, the proposed matheuristics exploits the advantages of exact methods in solving separately the CVRPTW, CLP, and TU selection problems, while taking advantage of heuristics to compute a solution for the combined logistic problem. This allows to overcome the limitations that emerged from the literature review, that is, on the one hand, to avoid simplifying assumptions for the mathematical formalization of the 3L-CVRPTW, which would lead to the representation of only a limited set of logistic rules; and on the other hand, to avoid addressing the problem with heuristic approaches only, which would disregard the mathematical programming model of the problem ad consider ad hoc solutions.

The proposed algorithm is composed of four phases and is suitable for the 3L-CVRPTW solution considering the following assumptions: multiple types of TUs with limited capacity are available, the number of available TUs for each type is unlimited, a single TU can include the cargo of multiple clients, the cargo of a client can exceed the capacity of the available TUs and in this case, it must be split among multiple TUs, for each route a client is visited only once in the related time window, a loading plan can be acceptable only if its fill ratio is higher than or equal to a given threshold.

Given the set of Table6.6, the four phases of the algorithm are the following:

Pre-processing phase: This phase consists in the initialization of the setting parameters of the matheuristcs and in the solution of a TU selection problem, aimed at both partitioning the cargo of a client over multiple TUs (if necessary) and to fasten the solution of the CVRPTW. The first section in **Algorithm - Pre-processing phase** summarizes the pre-processing phase. After the initialization of the setteing parameters. in the algorithm, for each client i, if the corresponding cargo exceeds the admissible weight or volume of the smallest available TU, the TU selection MILP problem (6.34)-(6.38), detailed below, is solved. Thus the best type of TU to be used and

Table 6.6: List of sets of the matheuristics.

Name	Description
$\mathcal{N}$	Set of clients
$\mathcal{B}$	Set of bins
$\mathcal{M}$	Set of TUs
$\tilde{\mathcal{B}}$	Set of bins' positions inside a TU
$\mathcal{P}$	Set of TUs, each assigned to a route in the pre-processing phase
$\mathcal{C}$	Set of clusters
$\mathcal{F}$	Set of TUs, each assigned to a load and route plan
f	Auxiliary set of TUs, each assigned to a load and route plan

the eventual partitioning of the delivery among multiple TUs (also of different types) is computed. The obtained solution $\mathcal{P}_i$ is satisfactory if the weight or volume assigned to the resulting TUs is at least equal to 90% of the maximum admissible value. Otherwise, the solution is discarded and the set of clients $\mathcal{N}$ is updated to $\mathcal{N}'$ where client i is replicated and the corresponding bins $\mathcal{B}_i$ are halved.

The TU selection problem computes the proper number and type of TUs to be assigned to a client i depending on the volume and weight of the corresponding cargo. Given the common variables and parameters of Table 6.2 and Table 6.3, the binary variables n^k and p_b^k which respectively indicate if TU k is used or not and if bin b is inside or outside TU k, and the parameters $\mathrm{D}_{0,i,(N+1)}$ representing the total length of route initial depot-client i-final depot, g_1 and g_2 respectively the weights of the base cost term and the variable cost of the objective function, the problem is formulated as follows:

$$\min(\mathrm{g}_1 \sum_{k=1}^{\mathrm{M}} \mathrm{C}^k n^k + \mathrm{g}_2 \sum_{k=1}^{\mathrm{M}} \mathrm{V}^k \mathrm{D}_{0,i,(\mathrm{N}+1)}\ n^k) \tag{6.34}$$

subject to:

$$\sum_{b=1}^{\mathrm{B}_i} p_b^k \leq n^k \mathrm{B}_i,\ \forall k \tag{6.35}$$

$$\sum_{k=1}^{\mathrm{M}} p_b^k = 1,\ \forall b \tag{6.36}$$

$$\sum_{b=1}^{\mathrm{B}_i} \mathrm{w}_b p_b^k \leq \mathrm{Q}_k,\ \forall k \tag{6.37}$$

$$\sum_{b=1}^{\mathrm{B}_i} \iota_b \chi_b \gamma_b\ p_b^k \leq \mathrm{W}_k \mathrm{L}_k \mathrm{H}_k,\ \forall k \tag{6.38}$$

Note that the objective function (6.34), to be minimized, is the sum of the base and variable costs of the TUs to be assigned to client i. Constraints (6.35) impose that the bins assigned to the transport units do not exceed the total number of bins of client i. Constraints (6.36) impose that a bin is placed in only one TU, and finally, constraints (6.37)-(6.38) impose that the total dimension and weight of the bins assigned to each TU do not exceed its capacity. This

problem, as already mentioned, is executed only for clients whose cargo exceeds the volume or supported weight of the smallest available TU. The problem aims at providing as output the best combination between the splitting of bins and the most convenient TUs for serving the client.

If the cargo of the client exceeds the capacity of the smallest TU, and the problem solution guarantees an occupation of the TU higher than 90%, the selected TU is added to the set $\mathcal{P}$ and used directly in the container loading phase without considering that address in the vehicle routing phase (to this aim the sets $\mathcal{N}$ and $\mathcal{B}$ are updated by removing the client and related bins). Otherwise, the algorithm updates the sets $\mathcal{N}'_i$ and $\mathcal{B}'_i$ (input of the next phase) by doubling the client in the set $\mathcal{N}'_i$ and changing the assignment bin-client in the set $\mathcal{B}'_i$ so as to let to the vehicle routing problem to assign to the specific client two different TUs.

Algorithm

Pre-processing phase

Initialize M, N, B, g_1, g_2, f_1, f_2, f_3, $\mathcal{N}$, $\mathcal{B}$, $\mathcal{M}$, T_{max}
Set $\mathcal{P} = p = \emptyset$
$Q_{min} = \min Q_k \quad Vol_{min} = \min(L_k W_k H_k)\ \forall k \in \mathcal{M}$
for i =1:N **do**
  Take $\mathcal{B}_i \subseteq \mathcal{B}$
  if $\sum_{b=1}^{B_i} w_b \geq Q_{min}$ or $\sum_{b=1}^{B_i} \chi_b \iota_b \gamma_b \geq Vol_{min}$ **then**
    Given $\mathcal{N}_i$, $\mathcal{B}_i$, $\mathcal{M}$ solve problem (6.34) - (6.38) for $\mathcal{P}_i$
    if $\sum_{b=1}^{|B_i|} w_b \geq 0.9\ Q_k$ or
$\sum_{b=1}^{|B_i|} \chi_b \iota_b \gamma_b \geq 0.9\ L_k W_k H_k \forall k \in \{1, \ldots, |\mathcal{P}_i|\}$ **then**
      $\mathcal{P} = \mathcal{P} \cup \mathcal{P}_i$, $\mathcal{N}' = \mathcal{N} \setminus \mathcal{N}_i$, $\mathcal{B} = \mathcal{B} \setminus \mathcal{B}_i$
    else
      $\mathcal{N}'_i = \mathcal{N}_i$, $\mathcal{N}' = \mathcal{N} \cup \mathcal{N}'_i$ $\mathcal{B} = \mathcal{B} \setminus \mathcal{B}_i$
      Set $\mathcal{B}_i = \{b_z, z = 1, \ldots, |\mathcal{B}_i|/2\}$
        $\mathcal{B}'_i = \{b_z, z = (|\mathcal{B}_i| + 1)/2, \ldots, |\mathcal{B}_i|\}$
      $\mathcal{B} = \mathcal{B} \cup \mathcal{B}_i \cup \mathcal{B}'_i$
    end if
  end if
end for

Vehicle routing phase: the main objective of this phase is the definition of the preliminary routing plan for all TUs starting with the sets of $\mathcal{N}'$ and $\mathcal{B}$ updated in the pre-processing phase. The second section in **Algorithm - Vehicle routing phase** summarizes this phase. It begins with the definition of the distance and travel times matrices computed considering all the clients of the set $\mathcal{N}'$. Then clients are grouped by geographical area using a hierarchical clustering, based on the work by [12]. This operation has the objective of reducing the computational effort of the VRP problem without compromising the optimality of the solution. The number of clusters depends on the number of clients contained in $\mathcal{N}'$, it is iteratively calculated modifying the parameter of the clustering in order to have at least a certain number of addresses for each cluster. Then, for each of the groups of clients, $\mathcal{N}_c$ and related bins $\mathcal{B}_c$ obtained with the clustering, the problem (6.3)-(6.12) is solved to obtain the desired routes. At the end of the execution of this phase, for each of TU obtained, the algorithm verifies if its weight fill ratio (i.e., the percentage occupation of the TU over the maximum admissible weight) is at least equal to 50%, or if the volume fill ratio (i.e., the percentage occupation of the TU over the maximum admissible volume) is at least equal to 50%. All the TUs that satisfy this requirement are used next in the container loading phase, while the ones that do not satisfy this requirement are discarded, their clients (and consequently

their bins) are grouped together and used for the last execution of the problem (6.3)-(6.12). The final outcome of the vehicle routing phase is the set $\mathscr{P}_p$ containing all the TUs assigned to a route (and consequently the list of addresses and bins).

Algorithm

Vehicle routing phase

Compute $\mathrm{E}_{i,j}$,$\mathrm{D}_{i,j}$ $\forall\ i,j \in \{1,\ldots,|\mathcal{N}'|\}, i \neq j$
Group $a \in \mathcal{N}'$ with the hierarchical clustering
Given the obtained $\mathcal{C}$ set of $|\mathcal{C}|$ clusters
for $c = 1 : |\mathcal{C}|$ **do**
  $\mathcal{N}_c \subseteq \mathcal{N}'$ $\mathcal{B}_c \subseteq \mathcal{B}$ $\mathscr{P}_{|\mathcal{C}|} = \emptyset$
  Given $\mathcal{N}_c, \mathcal{B}_c, \mathcal{M}$ solve problem (6.3)-(6.12) for $\mathscr{P}_c$
  $\mathscr{P}_{|\mathcal{C}|} = \mathscr{P}_{|\mathcal{C}|} \cup \mathscr{P}_c$
end for
$\mathscr{P}'_{|\mathcal{C}|} = \mathscr{P}_{|\mathcal{C}|} \cup \mathscr{P}$, n= 0, $\mathscr{P}_n = \emptyset$
for $k = 1 : \mathscr{P}'_{|\mathcal{C}|}$ **do**
  if $\sum_{b=1}^{|B_k|} \mathrm{w}_b \leq 0.5\ \mathrm{Q}_k$ or
    $\sum_{b=1}^{|B_k|} \chi_b \iota_b \gamma_b \leq 0.5\ \mathrm{L}_k \mathrm{W}_k \mathrm{H}_k$ **then**
    $\mathscr{P}_n = \mathscr{P}_n \cup \mathscr{P}_k$,
  end if
end for
$\mathscr{P}''_{|\mathcal{C}|} = \mathscr{P}'_{|\mathcal{C}|} \setminus \mathscr{P}_n$, $\mathcal{N}_n \subseteq \mathcal{N}'$ $\mathcal{B}_n \subseteq \mathcal{B}$
Given $\mathcal{N}_n, \mathcal{B}_n, \mathcal{M}$ solve problem (6.3)-(6.12) for $\mathscr{P}_n$
$\mathscr{P}_p = \mathscr{P}''_{|\mathcal{C}|} \cup \mathscr{P}_n$

Container loading phase: this phase aims at finding the optimal configuration of the bins inside each TU, once the set $\mathscr{P}_p$ containing all TU-clients assignment has been collected from the previous phases. The obtained TUs are given one by one as input to the CLP solver based on problem (6.13)-(6.33). This phase is executed in the same loop of the post-processing phase (described below) and detailed in the third section of **Algorithm -Container loading and post-processing phases**, with the parameters explained in Table 6.2.

Post-processing phase: this phase is used to manage the different situations in which the produced solutions of the container loading phase are not satisfactory (e.g., empty TUs' grouped in the container loading phase) or also when the CLP solver is not able to find a solution due to excessive computational time (T_{max}) or also positioning constraints. When these exceptions happen, the CLP solver is executed only for those TUs reducing iteratively the number of bins given as input (and consequently the set $\mathscr{B}_k$), starting from the one with the highest priority (so the nearest to the TU's door) until a solution is found. In this case two options can occur: the first client to be served has a number of bins such that even with the removed ones there are some bins of that client still in the TU, while in the second case all the bins of the client are removed and so the related stop is deleted from the route assigned to that TU. Then for both the options, the removed bins, their addresses and the set with the empty TUs grouped in the container loading phase are recombined for the last time so as to find a new route firstly executing the VRP's solver and then the loading plan for the CLP's one. The final output of this phase, which coincides with the output of the whole algorithm, is the set $\mathscr{F}$ of TUs, each assigned to a load and route plan.

Algorithm

Container loading and post-processing phases

for $k = 1 : |P_p|$ **do**
  $\mathcal{n}_k \subseteq \mathcal{n}'$ $\mathcal{B}_k \subseteq \mathcal{B}$
  Given $\mathcal{n}_k, \mathcal{B}_k, \mathcal{P}_k$ solve problem (6.13)-(6.33)
    for f_k,$\tilde{\mathcal{B}} \subseteq \mathcal{B}_k$
  if $\tilde{\mathcal{B}} \neq \mathcal{B}_k$ or $t >$ T$_{max}$ **then**
    X=0.9
    while Solution is not found **do**
      W=$|\mathcal{B}_k|$ Y=$X * W$
      Set $\mathcal{B}_{k'} = \{b_y, y = 1, \ldots, Y\}$
        $\mathcal{B}_{k''} = \{b_z, y = (Y + 1), \ldots, W\}$
      Given $\mathcal{n}_k, \mathcal{B}_{k'}, \mathcal{P}_k$ solve problem (6.13)-(6.33)
        for f_p,
      X=X-0.1
    end while
  end if
  $\mathcal{F}_k = \mathcal{F}_k \cup f_k$
end for
Extract $\mathcal{n}_{k'} \subset \mathcal{n}_k$ which contains all the addresses
  associated to $\mathcal{B}_{k''}$
Extract $\mathcal{F}'_k$ such that $\sum_{b=1}^{\mathcal{B}_m} w_b \geq 0.5\ Q_k$ or
  $\sum_{b=1}^{\mathcal{B}_m} \chi_b \iota_b \gamma_b \geq 0.5 \mathrm{L}_k \mathrm{W}_k \mathrm{H}_k \forall\ k \in \mathcal{P}_p$
$\mathcal{F}_F = \mathcal{F}_k \setminus \mathcal{F}'_k$
Given $\mathcal{A}_{p'}, \mathcal{B}_{p''}, \mathcal{T}$ solve problem (6.1)-(6.12) for $\mathcal{P}_f$
Given $\mathcal{A}_{p'}, \mathcal{B}_{p'}, \mathcal{P}_f$ solve problem (6.13)-(6.33) for f_f
$\mathcal{F} = \mathcal{F}_F \cup f_f$

6.4 Experimental Results

In this section, first, the proposed method is tested on an extensively used literature's benchmark. Then, it is applied on a real database provided by the Italian company E80 Group. The tests were conducted in Matlab2020a, on a PC equipped with an 2.20 GHz Intel Corei7-8750H CPU and 32 GB RAM.

6.4.1 Tests on a Literature Benchmark

The proposed method is tested on one of the most complete literature dataset for the routing and container loading problem presented in [13]. This dataset is composed by 27 Euclidean Capacitated Vehicle Routing Problem instances, containing a number N of clients that ranges from 15 to 100 with only one depot, and a number B of bins that varies from a minimum of 32 to a maximum of 198, each bin is assigned a fragility index. For each instance, only one type of TU is considered, with dimensions W = 2500 cm, H = 3000 cm, and L = 6000 cm and a maximum number of available TUs is set. It has to be highlighted that the considered dataset does not include some of the parameters necessary for the setting of the proposed matheuristics, i.e., time windows, cargo balancing, load bearing, stability and bins' positioning constraints. Consequently, the related constraints and logic rules present in the proposed method were disabled. Table 6.7 shows the results achieved with the two methods, i.e., the proposed matheuristics and the method in [13]. In columns II to IV it is reported the identification code of the instance, the number of clients N and the number of bins B. In columns V-XI the achieved results for each

instance are reported, i.e., the number m_1 and m_2 of filled TUs obtained respectively with the matheuristics and by the authors of [13], the total length of the routes assigned to the TUs by the proposed approach (Z1) and by that used [13] (Z2), the percentage of improvement obtained with the proposed model (Δ) in terms of travel distance. The results demonstrate that the proposed method uses a lower number of TUs than the ones computed in [13] in 88.88% of instances, and moreover achieves a better value of the objective function for the 74% of instances, with an average improvement of 7.0%.

Table 6.7: Comparison of the matheuristic approach with the literature benchmark.

Instance	N	B	m_1	m_2	Z1 [km]	Z2 [km]	Δ [%]
E016-03m	15	32	5	5	320.0	316.3	-1.2
E016-05m	15	26	5	5	320.0	350.6	8.7
E021-04m	20	37	5	3	316.6	447.7	29.3
E021-06m	20	36	4	6	346.6	448.5	22.7
E022-04g	21	45	3	7	421.0	464.2	9.3
E022-06m	21	40	3	6	404.9	504.5	19.7
E023-03g	22	46	3	6	673.5	831.7	19.0
E023-05s	22	43	4	8	842.7	871.8	3.3
E026-08m	25	50	4	8	570.8	666.1	14.3
E030-03g	29	62	5	10	898.9	911.2	1.4
E030-04s	29	58	5	9	803.4	819.4	2.0
E031-09h	30	63	5	9	474.3	615.6	23.0
E033-03n	32	61	6	9	3,022.5	2,928.0	-3.2
E033-04g	32	72	6	11	1,471.2	1,559.6	5.7
E033-05s	32	68	8	10	1,790.3	1,452.3	-23.3
E036-11h	35	63	5	11	887.7	707.8	-25.4
E041-14h	40	79	6	14	647.5	920.9	29.7
E045-04f	44	94	8	14	1,108.3	1,400.5	20.9
E051-05e	50	99	9	13	893.3	871.3	-2.5
E072-04f	71	147	17	20	759.2	732.1	-3.7
E076-07s	75	155	11	18	1,122.5	1.275.2	12.0
E076-08s	75	146	13	19	1,243.4	1,278.0	2.7
E076-10e	75	150	14	18	1,187.8	1,258.1	5.6
E076-14s	75	143	13	18	1,158.5	1.307.1	11.3
E101-08e	100	193	19	24	1,698.8	1,570.7	-8.2
E101-10c	100	199	19	28	1,746.3	1,847.9	5.5
E101-14s	100	198	15	25	1,535.0	1,747.5	12.2

6.4.2 Tests on a Real Case Study

Here the results obtained with a real dataset provided by the Italian logistic company Elettic80 are presented. In particular, the dataset includes 52 clients distributed all over Italy, 299 bins with different heights and weights but the same base area (i.e., the EU standard pallet with dimension 1200x800 mm), and 3 different types of TUs ideally infinite. The dataset has been then mixed and combined producing different instances each one characterized by a different number of clients, bins, and geographycal distibution of the clients. The algorithm tested is the one proposed in Section 6.3, and in order to have computational times that can be easily adapted to the industrial needs for the loops involved in the Preprocessing Phase, the Vehicle Routing

Phase and the Container Loading Phase are executed in parallel using the Parallellization Toolbox provided by Matlab that allows to run more than one instance of the solver simultaneously. This is possible because each loop execute separately the relative mathematical model for different elements of the dataset, and so each iteration is completely independent to the other. Table 6.8 reports the results obtained with the proposed matheuristics on the real dataset. In particular, column I reports the identification code of the instance, columns II and III respectively report the number N of clients and the number B of bins. Column IV reports the number of filled TUs, and the computational time T_{ex}. The weight fill ratio of the TUs achieved in all the tests is on average 78.71 % of the supported weight, while the volumetric fill ratio achieved in all the tests is on average 47.65 % of the admissible volume. This depends on the fact that the weight of the bins in the dataset is higher than their volume, so in many cases the total weight transported reaches the 80%-90% of the total supported weight with just few bins. For what concerns the total travelled distance, it ranges from a maximum of 8.012,263 meters to a minimum of 1.012,263 meters. Finally, as for T_{ex}, it ranges from 1.10 to 3.90, which are appreciable values, considered the dimensions of the instances.

Table 6.8: Real case study results.

Instance	N	B	M	T_{ex}[s]
A1	52	299	18	3,896
A2	25	131	5	2,324
A3	30	210	7	1,275
A4	43	235	10	2,029
A5	15	62	12	1,947
A6	17	110	5	2,337
A7	15	55	5	1,096

6.5 Conclusion

In this work proposes an innovative matheuristics to efficiently solve the integrated vehicle routing and container loading problem for the logistic sector. In particular, it addresses the three dimensional loading capacitated vehicle routing problem with time windows. In the related literature just a few contributions address this combined problem with logistic constraints and consider only a restricted set of logistic constraints. On the contrary, The proposed method aims at supporting the external logistic sector in route and load planning operations including a large set of realistic logistic constraints. The tests demonstrate the effectiveness of the proposed solution both in comparison with a literature benchmark and with a real dataset. Future works will consider the possibility to manage the delivery in real time in order to provide the driver with updated routes in case of unexpected events.

References

[1] Tresca, G., Cavone, G., Carli, R., Cerviotti, A., and Dotoli, M., "Automating bin packing: A layer building matheuristics for cost effective logistics," *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 3, pp. 1599–1613, 2022.

[2] Golden, B. L., Raghavan, S., Wasil, E. A., *et al.*, *The vehicle routing problem: latest advances and new challenges*. Springer, 2008, vol. 43.

[3] Cavone, G., Carli, R., Troccoli, G., Tresca, G., and Dotoli, M., "A milp approach for the multi-drop container loading problem resolution in logistics 4.0," in *2021 29th Mediterranean Conference on Control and Automation (MED)*, 2021, pp. 687–692. DOI: 10.1109/MED51440.2021.9480359.

[4] Dantzig, G. B. and Ramser, J. H., "The truck dispatching problem," *Management science*, vol. 6, no. 1, pp. 80–91, 1959.

[5] Goldberg, D. E., Lingle, R., *et al.*, "Alleles, loci, and the traveling salesman problem," in *Proceedings of an international conference on genetic algorithms and their applications*, Carnegie-Mellon University Pittsburgh, PA, vol. 154, 1985, pp. 154–159.

[6] Bai, R., Chen, X., Chen, Z., *et al.*, "Analytics and machine learning in vehicle routing research," *CoRR*, vol. abs/2102.10012, 2021.

[7] Moura, A., "A multi-objective genetic algorithm for the vehicle routing with time windows and loading problem," in *Intelligent decision support*, Springer, 2008, pp. 187–201.

[8] Pollaris, H., Braekers, K., Caris, A., Janssens, G. K., and Limbourg, S., "Vehicle routing problems with loading constraints: State-of-the-art and future directions," *OR Spectrum*, vol. 37, no. 2, pp. 297–330, 2015.

[9] Dotoli, M. and Epicoco, N., "A vehicle routing technique for hazardous waste collection," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 9694–9699, 2017.

[10] Wäscher, G., Haußner, H., and Schumann, H., "An improved typology of cutting and packing problems," *European journal of operational research*, vol. 183, no. 3, pp. 1109–1130, 2007.

[11] Bortfeldt, A. and Wäscher, G., "Container loading problems: A state-of-the-art review," *Working Paper Series*, 2012.

[12] Johnson, S. C., "Hierarchical clustering schemes," *Psychometrika*, vol. 32, no. 3, pp. 241–254, 1967.

[13] Gendreau, M., Iori, M., Laporte, G., and Martello, S., "A tabu search algorithm for a routing and container loading problem," *Transportation Science*, vol. 40, no. 3, pp. 342–350, 2006.

Chapter 7

A Matheuristic Approach for Delivery Planning and Dynamic Vehicle Routing in Logistics 4.0

VII

Abstract

In distribution logistics, the planning of vehicles' routes and vehicles' loads is traditionally managed separately, despite this activity being correlated. This often leads to various re-designs to make the routes and load plans compatible with each other and applicable in practice. Moreover, the planned routes, which are static by definition, cannot always cope with unexpected events. Traffic congestion, vehicle failures, adverse meteorological conditions, and further undesired events can make the planned routes inapplicable and require *vehicles' re-routing*. This results in lower service levels, undesired delays, and higher costs for logistics companies. With the aim of overcoming the above limitations, this work proposes a novel algorithm that jointly solves the problem of *delivery planning* and *dynamic vehicle routing* to automate the delivery process in a logistics 4.0 perspective. The presented algorithm includes two different phases: the static phase, which is executed offline and in advance with respect to the delivery day, and the dynamic phase, which is executed in real-time to cope with unexpected events during the delivery. For the first phase, a matheuristics approach is defined to efficiently solve the combined vehicle routing and loading problems. Differently, for the second phase, a genetic algorithm is proposed to re-route vehicles in real-time, considering both the redefinition in real-time of the nominal trip and/or of the sequence of the customers to be visited. The algorithm is tested both on a literature benchmark and on a real dataset provided by an Italian logistics company. The obtained results show that, on the one hand, the proposed algorithm can automatically provide feasible solutions that minimise travel costs, total travelled distance, and empty space on the vehicles; on the other hand, it can ensure in real-time effective re-routing solutions in case of unexpected events occurring during the delivery.

Contents

7.1 Introduction

Distribution logistics is one of the four sub-sectors of logistics, i.e., purchase, production, distribution, and after-sales logistics that generates the highest percentage of logistics operations

costs [1]. On the one hand, this mostly depends on the absence of a holistic standardized and automated approach to the resolution of the related decision problems, i.e., (1) the choice of the transportation mode, (2) the packing of goods, (3) the planning of vehicle loads, and (4) the definition of optimal vehicle routes [2]. On the other hand, this derives from the inability of the offline generated plans to limit undesirable costs resulting from unforeseen events that can occur during the deliveries [1]. This chapter focuses on two of the four crucial decision problems of distribution logistics, i.e., (3) and (4), for which the related literature abounds with contributions, but mainly for their offline planning only. These problems are typically solved separately as Container Loading Problem (CLP) and Vehicle Routing Problem (VRP), and respectively aim at defining loading plans for the Transport Units (TUs) devoted to the deliveries and at planning the corresponding feasible routes by optimising a proper performance indicator. Only recently, the scientific literature has focused on the combination of the loading and routing problems in distribution planning to reduce the related costs [2]. Among the various categories of VRP with loading constraints, the most interesting for practical purposes is the so-called three-dimensional Loading VRP (3L-VRP). In particular, the most comprehensive definition in this regard is the three-dimensional Loading Capacitated VRP with Time Windows (3L-CVRPTW) presented in [3] and subsequently extended in [4], which aims at finding feasible delivery routes, while taking into account capacity limitations of transportation means and time windows for the deliveries.

Although the existing contributions are effective in computing efficient distribution plans, such resulting plans are barely able to address unexpected events during the delivery, such as traffic congestion, weather conditions, etc. Consequently, some approaches have been proposed in the related literature to overcome such limitations. In particular, the majority of contributions regard the Dynamic Vehicle Routing Problem with Time Windows (DVRPTW) [5]. In such a problem the delivery plan is changed in real-time based on the variable arrival of clients' orders. Nevertheless, as highlighted in [6], only a few works have addressed the more general case of dynamic vehicle routing, in which deliveries are updated in real-time in response to external disturbances on the environment state. The reason for this lack of contributions lies in a low level of technology advancement that limits logistics companies in monitoring and re-routing in real-time their vehicles fleet [7].

With the aim of overcoming the above limitations and supporting the automation of the distribution process in the Logistics 4.0 perspective ([8], [9]), in this chapter, it is proposed a matheuristic approach for the integrated management of delivery planning and dynamic vehicle routing. The proposed approach includes two phases: the static phase, which is devoted to the offline resolution of the 3L-CVRPTW, i.e., determining the routing and loading plans; the dynamic phase, which is devoted to the real-time resolution of the Vehicle Routing with Time Windows (VRPTW) in case of unexpected events, i.e., updating dynamically the routing plans. It is highlighted that this work is based on the preliminary results presented by some of the authors in [10], where the baseline for the static phase is presented. The algorithm is tested both over a literature benchmark and a real dataset provided by an Italian logistics company. The obtained results show that, on the one hand, the proposed approach can provide feasible solutions that minimise travel costs, total travelled distance, and empty space on the vehicles; on the other hand, it can ensure in real-time effective re-routing solutions in case of unexpected events occurring during the delivery.

Despite the rich state of the art on CLP, VPR, and DVRP, very few research studies pay attention to the definition of a holistic standardized and automated approach to their combined solution. In fact, for what concerns the combination of the CLP with the VRP, only a few contributions are presented in the literature, i.e., [3],[11] and [12], and their formulations do not take into account realistic logistic requirements, such as the balancing of the cargo and its vertical and horizontal stability. In some other industrial approaches like [13]-[14], the most of the times

considers most of the features contained in this work but lacks of some others (e.g., such as in' stacking and stability, reachability, allocation-connectivity and allocation-separation), and do not consider some innovative features such as the selection of the transport unit according to the cost and the managing of the dynamic occurrence during the shipments (i.e., traffic congestions, variation in the travel time and in the service time). In addion, it has to be noticed that the novelty of the proposed method goes beyond the innovative formulation of integrating the 3L-VRPTW and DVRP. It also includes the implementation of a new matheuristic approach that enables efficient and automated management of the entire delivery process. This management includes analyzing the shipment list, determining the most suitable bin configuration within transport units (TUs), planning the corresponding routes, and handling unforeseen events during deliveries. In fact, the growing attention from the scientific community to matheuristics (as seen in the paper by [15]) demonstrates the effectiveness of this approach. Furthermore, as for the VRP planning, time-dependent travel times have received some attention in the literature [16] [17] [18] [19] considering a static approach: travel times are typically considered stochastic, only computed once, and generally not subject to major disturbances (which could instead occur, for example in case of accidents). Regarding the dynamic VRP, the majority of literature contributions consider customer requests to be the dynamic element of the problem [20] [21] [22] [23]. In contrast with the state of the art, this chapter proposes a novel algorithm for the integrated solution of delivery planning combining offline VRP and CLP planning and DVRP, to improve the efficiency of external logistics in a Logistics 4.0 perspective. In particular, the main contributions of this work are:

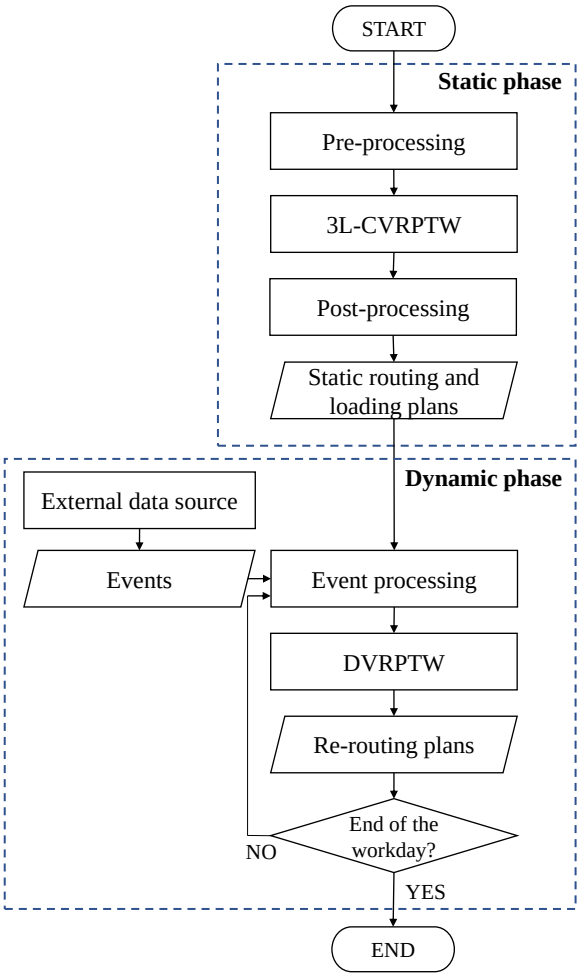
- the definition of a DVRPTW and of an event-driven heuristics that takes advantage of a genetic algorithm for the fast solution of the problem in real time. The event-driven heuristics is in charge of the re-routing and is triggered by disturbance events that might occur during the delivery, .e., undesired variations in travel time, service time, route distance, and arrival time of TUs. When the re-routing is triggered, the affected routes are re-optimized considering the re-ordering of clients in the planned routes, except for the first and final client, and the skip-stop of some clients in the route if the constraints on time windows are not fulfilled;
- the definition of a two-phase matheuristics for the joint management of the delivery planning and the dynamic vehicle routing based on the formulation of the 3L-CVRPTW and the DVRPTW. The first phase, namely the static phase, addresses the planning of the deliveries in terms of routes, number of TUs, and their loading configurations, taking into account logistic constraints that are barely considered in the literature (i.e., vertical and horizontal stability, load balancing, and LIFO ordering constraints). The second phase, namely the dynamic phase, addresses the dynamic vehicle routing in case of unexpected disturbances during the deliveries. The proposed method allows, in a short computational time, the proper and automatic management of the delivery process from the analysis of the shipment list to the definition of the most suitable configuration of bins in the TUs and the related routes as well as the management of unexpected events during the deliveries;
- the implementation of extensive computational tests to evaluate the performance of the proposed algorithm using both realistic data-sets drawn from the literature and real data-sets from an Italian logistic company. In particular, it is shown the effectiveness of the proposed methodology in managing the delivery planning and tackling unexpected events affecting the TUs' travels.

The remainder of the chapter is organised as follows. Section 7.2 describes in detail the proposed algorithm for the combined solution of delivery planning and dynamic vehicle routing.

Section 7.1 details the formulation of the DVRPTW, while Section 7.4 presents the heuristics-based solution for the DVRPTW. Section 7.5 presents and discusses the results obtained with the proposed algorithm. Finally, Section 7.6 concludes the chapter.

7.2 The Delivery Planning and Dynamic Vehicle Routing Matheuristics

Figure 7.1: High-level flow chart of the proposed algorithm.



The proposed two-phase matheuristics is devoted to two macro functions: 1) the offline computation of routing and loading plans; 2) the re-routing of the TUs in case of unexpected events during the delivery execution. Consequently, the algorithm can be executed each time a novel set of deliveries has to be managed and is composed of the **Static phase** and the **Dynamic phase**, which are executed sequentially. The high-level flowchart of the algorithm is reported in Fig. 7.1 where the two sequential phases are highlighted by dashed squares. It is highlighted that the two different phases are strictly correlated. In fact, the output of the static phase is used

as the initial solution for the dynamic phase, which is in charge of TUs' re-routing in case of unexpected events. Details on each phase are provided in the following subsections.

7.2.1 Static Phase

This phase regards offline delivery planning. The algorithm receives as input the list of deliveries, the related goods, and the available TUs, as well as its setting parameters, and provides as output feasible static routing and loading plans (see Fig. 7.1). It is highlighted that, since this phase is performed several days in advance with respect to the actual deliveries, the computational time is not a primary concern. The method is composed by three main sub-phases, i.e., *pre-processing, 3L-CVRPTW, and post-processing*, and combines the exact solution of three MILP problems, used across the whole algorithm that take into account basic logistic constraints, and heuristic methods that allow to include additional more complex logistic constraints while limiting the computational effort. It is highlighted that the algorithm was presented in Chapter 6 and is composed of three sub-phases, i.e., *pre-processing, 3L-CVRPTW, and post-processing.* thus, for a matter of brevity, here it is reported only the objective function of each problem and a high-level description for the set of related constraints, while the flowcharts of each sub-phase of the algorithm are reported in Fig.s 7.2, 7.3, 7.4.

Table 7.1: List of sets of the static phase.

Name	Description
	Input Sets
$\mathcal{A}$	Set of clients
$\mathcal{B}$	Set of bins
$\mathcal{T}$	Set of TUs
	Auxiliary Sets
$\mathcal{A}'$	Auxiliary set of clients
$\mathcal{A}_k$	Auxiliary set of clients of k-th TU
$\mathcal{B}'$	Auxiliary set of bins
$\mathcal{B}'_i$	Auxiliary set of bins of i-th client
$\mathcal{B}_k$	Auxiliary set of bins of k-th TU
$\mathcal{P}', \mathcal{P}'', \mathcal{P}'''$	Auxiliary sets of route plans
$\mathcal{F}''$	Auxiliary set of unsuccessful load plans. Each element is a triplet including the ordered set of client/s assigned to the TU, the TU, and the related bins
$\mathcal{F}'''$	Auxiliary set of load plans. Each element is a triplet including the ordered set of client/s assigned to the TU, the TU, and the configuration of the related bins
$\mathcal{C}$	Set of clusters
	Output Sets
$\mathcal{P}$	Output set of the pre-processing sub-Phase. Each element is a triplet including a client, the assigned TUs, and related bins
$\mathcal{F}'$	Output set of the algorithm. Each element is a triplet including the ordered set of client/s assigned to the TU, the TU, and the configuration of the related bins

Pre-processing sub-phase: this sub-phase performs a first check on the data to prevent the

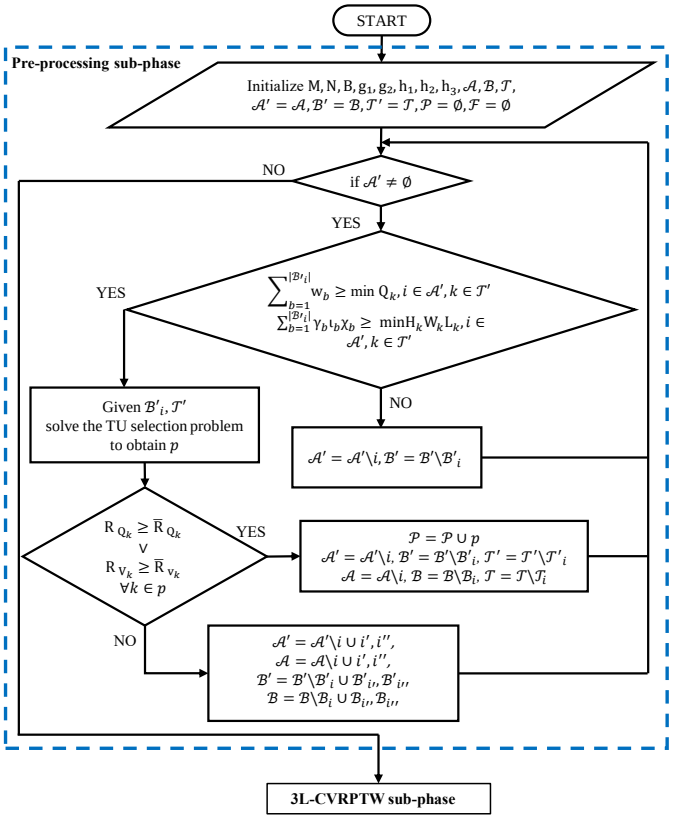
occurrence of infeasible and incomplete solutions and simplify the resolution of the subsequent 3L-CVRPTW. The corresponding flow-chart reported in Fig. 7.2 shows that first an empty set $\mathscr{P}$ is defined, whose elements p are ordered triplets $(i, \mathscr{T}_i, \mathscr{B}_i)$ of client $i \in \mathscr{A}$, the related TUs $\mathscr{T}_i \in \mathscr{T}$, and the corresponding bins $\mathscr{B}_i \in \mathscr{B}$, and the auxiliary sets to be used in this sub-phase $\mathscr{A}' = \mathscr{A}$, $\mathscr{B}' = \mathscr{B}$, $\mathscr{T}' = \mathscr{T}$. Then, for each i-th client in $\mathscr{A}'$ only if the corresponding cargo exceeds the admissible weight or volume of the smallest TU among the M available ones, the TU selection problem is solved, otherwise, the subsequent 3L-CVRPTW sub-phase is executed. Basically, when clients have a small cargo, it is more appropriate to combine it with other clients' cargo, while when the cargo is larger than the smaller TU, then it is appropriate to select the most convenient TU/s to be used. When the TU selection problem must be solved, it requires the sets $\mathscr{B}'_i$, including the bins of the i-th client, and $\mathscr{T}'$, including the available TUs. The corresponding obtained solution p is then further evaluated. Only if the weight or volume assigned to each k-th TU of p, i.e., $\mathrm{R}_{\mathbf{Q}k}$ and $\mathrm{R}_{\mathbf{V}k}$, is at least equal to the desired reference percentage $\bar{\mathrm{R}}_{\mathbf{Q}k}$ and $\bar{\mathrm{R}}_{\mathbf{V}k}$ of its maximum admissible value, the solution is considered admissible and included in the set $\mathscr{P}$, the client i is removed from the sets $\mathscr{A}, \mathscr{A}'$, the corresponding bins $\mathscr{B}_i$, $\mathscr{B}'_i$ are removed from the sets $\mathscr{B}, \mathscr{B}'$, and the assigned TUs $\mathscr{T}_i$, $\mathscr{T}'_i$ are removed from the sets $\mathscr{T}$, $\mathscr{T}'$. Otherwise, the solution is discarded and in the sets of clients $\mathscr{A}$ and $\mathscr{A}'$ the client i is substituted with i' and i''. Similarly, in the sets of bins $\mathscr{B}$ and $\mathscr{B}'$, the set of bins of client i is substituted with $\mathscr{B}'_{i'}$ and $\mathscr{B}'_{i''}$. Note that i' and i'' are replicates of client i, while $\mathscr{B}_{i'}$ and $\mathscr{B}_{i''}$ include respectively half bins of $\mathscr{B}_i$. The objective function of the TU selection problem is formulated as follows and the variables and parameters are shown in Table 7.2:

$$\min(\mathrm{g}_1 \sum_{k=1}^{\mathrm{M}} \mathrm{C}^k n^k + \mathrm{g}_2 \sum_{k=1}^{\mathrm{M}} \mathrm{V}^k \mathrm{d}_{0,i,(\mathrm{N}+1)}\ n^k) \tag{7.1}$$

The aim is to minimize the weighted sum of the base and variable costs of the TUs to be assigned to client i (respectively with the weights g_1 and g_2 whose sum is 1) while fulfilling a set of specific constraints (see [9]): a bin must be placed in only one TU; the bins assigned to a TU must not exceed the total number of bins of client i; the total volume and weight of the bins assigned to each TU must not exceed its capacity.

Thus, the outputs of this sub-phase are the updated sets $\mathscr{A}, \mathscr{B}$, and the set $\mathscr{P}$, which is composed of the triplets p including the considered client, the best type of TUs to be used, and the corresponding bins. It is highlighted that for each element of $\mathscr{P}$, each TU is assigned only to a single client, thus the corresponding route is uniquely defined.

Figure 7.2: Flowchart of the pre-processing sub-phase.



3L-CVRPTW sub-phase: in this sub-phase the 3L-CVRPTW is solved by combining the solution of CVRPTW and CLP, the related flowchart is reported in Fig. 7.3. The first part of this sub-phase is the definition of a preliminary routing plan considering the sets $\mathcal{A}$ and $\mathcal{B}$ redetermined in the pre-processing sub-phase. The distance and travel times matrices, i.e., **D**, **E**, are computed for the clients included in $\mathcal{A}$ and then they are used to group geographically the clients using the hierarchical clustering (for details see [24]). The result of this operation is a variable number of clusters that changes according to the number of clients included in $\mathcal{A}$ with a variable number of clients for each cluster (whose range is set with a setting parameter). This operation allows for reducing the computational effort of the algorithm. As a matter of fact, after the clustering, the next step of this sub-phase is the execution for each cluster of clients $\mathcal{A}_c$ and related bins $\mathcal{B}_c$, of the CVRPTW problem to obtain the desired routes. The outcome is a set $\mathcal{P}'$ whose elements p' are triplets $(\mathcal{A}_k, k, \mathcal{B}_k)$, including $\mathcal{A}_k \in \mathcal{A}$ the ordered set of clients associated with the k-th TU, k the index of the TU, and $\mathcal{B}_k \in \mathcal{B}$ the bins associated with the k-th TU. The complete MILP is described in [9], its objective function (7.2), reported below, is composed of three different terms representing the base costs, the variable costs, and the time window delay, weighted respectively with the weights h_1, h_2, and h_3 (whose sum is one), given in input by the company in order to modulate the importance of each term according to their needs.

Table 7.2: Parameters and variables of the static phase.

Name	Description
Input Parameters	
M	Number of TUs
N	Number of clients
g_1	Weight of the base cost term of the pre-processing
g_2	Weight of the variable cost term of the pre-processing
h_1	Weight of the base cost term of the CVRPTW
h_2	Weight of the variable cost term of the CVRPTW
h_3	Weight of the delivery delay term of the CVRPTW
C^k	Base cost of TU k
V^k	Variable cost of TU k
$d_{0,i,(N+1)}$	Total length of route initial depot-client i-final depot
E	Travel time matrix
D	Distance matrix
$d_{i,j}$	Kilometers from client i to j
ι_b	Length of bin b
χ_b	Width of bin b
γ_b	Height of bin b
$w_{b,i}$	Weight of bin b
L_k	Length of TU k
W_k	Width of TU k
H_k	Height of TU k
Q_k	Maximum weight supported by TU k
$R_{\mathbf{Q}k}$ ($\bar{R}_{\mathbf{Q}k}$, $\bar{R}'_{\mathbf{Q}k}$)	Weight fill ratio (Reference) of the TU k (i.e., percentage of weight transported by the TU k)
$R_{\mathbf{V}k}$ ($\bar{R}_{\mathbf{V}k}$, $\bar{R}'_{\mathbf{V}k}$)	Volume fill ratio (Reference) of the TU k (i.e., percentage of volume occupied in the TU k)
X,Y	Percentage of bins considered in the post-processing phase (Y=1-X)
Output Variables	
k	Index of TUs
i	Index of clients
b	Index of bins
n^k	The TU k is used (1) or not (0)
$t^k_{i,j}$	The k-th TU runs from client i to client j (1) or not (0)
$t^k_{0,j}$	The k-th TU runs from the initial depot to client j (1) or not (0)
δ^k_j	Advance/delay of the k-th TU arrival at client j with respect to opening time A_j

The constraints regard the route feasibility, TUs' capacity, and time windows' violations.

$$\min_k (h_1 \sum_{k=1}^{M} \sum_{j=1}^{N} C^k t^k_{0,j} + h_2 \sum_{k=1}^{M} \sum_{i=1}^{N} \sum_{j=1}^{N} V^k d_{i,j}\ t^k_{i,j} + h_3 \sum_{k=1}^{M} \sum_{j=1}^{N} \delta^k_j) \tag{7.2}$$

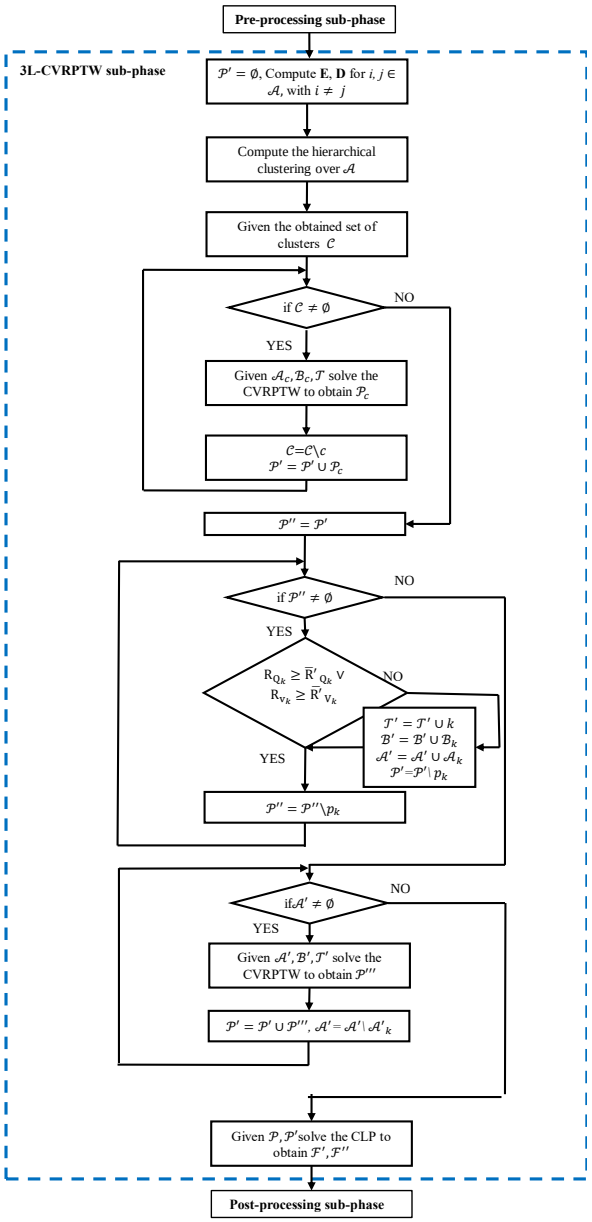
Thereafter, for each k-th TU assigned to a route and included in $\mathscr{P}'$, the algorithm verifies if the occupation of the TU over the maximum admissible weight (i.e. weight fill ratio $\mathrm{R}_{\mathbf{Q}k}$) and the maximum admissible volume (i.e. volume fill ratio $\mathrm{R}_{\mathbf{V}k}$) is in both cases higher than or equal to relative reference percentage values $\bar{\mathrm{R}}'_{\mathbf{V}k}$ $\bar{\mathrm{R}}'_{\mathbf{Q}k}$. Note that for such evaluation, the algorithm includes an auxiliary set $\mathscr{P}''$, which is equal to $\mathscr{P}'$. All the TUs that do not respect these values, their clients $\mathscr{A}_k$, and their bins $\mathscr{B}_k$) are grouped together in the sets $\mathscr{A}', \mathscr{B}', \mathscr{T}'$ respectively. The CVRPTW is solved again leading to the solution $\mathscr{P}'''$, to be added to the set $\mathscr{P}'$. The outcome of this last check is the complete delivery plan, i.e., the set $\mathscr{P}'$. In the second part of this sub-phase, for each element in $\mathscr{P}$ (obtained with the pre-processing sub-phase) and in $\mathscr{P}'$, i.e., a TU with the assigned client/s and related bins, the algorithm aims at finding the optimal configuration of the corresponding bins. For this purpose, the CLP detailed in Chapter 6, is solved for each k-th TU and the related $\mathscr{B}_k$ bins, using the variables and parameters of Table 7.2.

$$\min(\mathrm{L}_k\ \mathrm{W}_k\ \mathrm{H}_k - \sum_{b=1}^{|\mathscr{B}_k|} \iota_b\ \chi_b\ \gamma_b) \tag{7.3}$$

As shown by the objective function (7.3), the goal of the CLP problem is to minimize the unoccupied space inside each TU (calculated subtracting to the total volume of the k-th TU, the volume of the bins placed inside it), while finding the optimal configuration of the bins and fulfilling a wide set of constraints (see Chapter 6) related to the CLP feasibility (i.e., between two different bins, a bin can assume only one relative position with respect to the other one in the TU; the coordinates of two bins must not overlap and must not exceed the dimensions of the TU and the bin can rotate 90 degrees), positioning constraints that order the bins by height according to their fragility index and their priority index with an ascending order (i.e., the priority of the delivery to which the bin belongs, so that the ones with the highest values are the nearest to the door and so the first to be delivered), horizontal and vertical stability constraints, TUs' capacity constraints (both in terms of total volume loaded and maximum supported weight), and load balancing constraints, which ensure that the load is positioned inside the TU so that its center of mass lies in a bounded area inside the basis of the TU.

Finally the output of the 3L-CVRPTW sub-phase are the sets $\mathscr{F}'$ and $\mathscr{F}''$, whose elements are respectively $f' = (\mathscr{A}_k, k, \tilde{\mathscr{B}}_k)$ including the ordered set of client/s assigned to the k-th TU and the configuration of the related bins $\tilde{\mathscr{B}}_k$, and $f'' = (\mathscr{A}_k, k, \mathscr{B}_k)$ including the ordered set of client/s assigned to the k-th TU and the related bins without load plan. In particular, $\mathscr{F}'$ includes the satisfactory solutions in terms of load plans, while $\mathscr{F}''$ includes the unsatisfactory ones. In particular, the elements included in $\mathscr{F}''$ are triplets for which the assignment of the cargo to the TUs results from the solution of the CVRPTW, but for which the CLP does not succeed in defining a proper loading plan. Thus such elements must be rearranged and the cargo assigned to each TU with the CVRPTW must be revised to accomplish the CLP constraints.

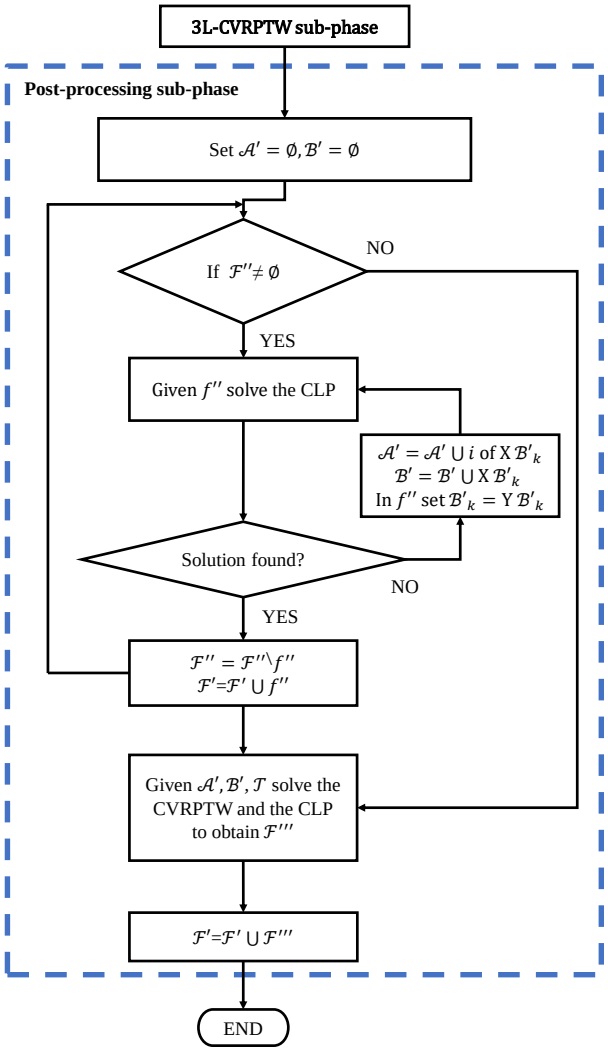
Figure 7.3: Flowchart of the 3LCVRPTW sub-phase.



Post-processing sub-phase: this sub-phase allows managing the unsatisfactory load plans obtained with the 3L-CVRPTW sub-phase. To this aim, the algorithm of this sub-phase, reported in Fig. 7.4, executes the CLP problem for each unsatisfactory solution included in $\mathcal{F}''$, and thus for each TU, reducing iteratively the set of bins associated with the TU until a solution is found. Consequently, given an element $f'' \in \mathcal{F}''$, first its bins are reduced by the X% (value set initially by the company) and the CLP is solved. Then, if a solution is found a new element is included in the set $\mathcal{F}'$ including the set of clients of the considered TU, the TU index, and the assigned bins with the related configuration. Conversely, the cargo is iteratively reduced by X%, until a solution to the CLP is found. These steps are then executed for each element of $\mathcal{F}''$. Finally, the

remaining clients and corresponding bins left out at each iteration are grouped and the CVRPTW and CLP are solved to obtain their load and route plans. The final output of the algorithm is then the set $\mathcal{F}'$ whose elements are the triplets including the ordered set of clients to be visited by a TU, the index of the TU, and the set of bins, with the related configurations, to be loaded in the TU.

Figure 7.4: Flowchart of the Post-processing sub-phase.



7.2.2 Dynamic Phase

This phase is performed online when the delivery plans are executed and the related TUs are dispatched. In this phase, it is assumed that the load plans have been already implemented and thus it is not possible to change the clients-to-vehicle assignments, as well as the number of vehicles. Conversely, the control actions that are allowed are: 1) re-ordering of clients in the planned routes, except for the first and final client; 2) skip-stop of some clients in the route if the constraints on time windows are not fulfilled. This phase consists of two main sub-phases that are iteratively executed during the working day. When the dynamic phase is started, first, the

Pseudocode - Dynamic phase

Initialization

Initialize M, N, $\mathcal{F}'$, h_1, h_2, h_3, V^k, C^k, $\mathbf{E}$, $\mathbf{F}$, $\mathbf{D}$, Σ_D, Σ_E, Σ_F

Event processing sub-phase

Given t and t_{max} (respectively current and maximum computation time)
Given $\mathbf{D}_t$, $\mathbf{E}_t$ and $\mathbf{F}_t$, matrices containing the new parameters
Given L_A, set of detected new vehicle arrivals
$\Delta_\mathbf{D} = \mathbf{D}_t - \mathbf{D}$
$\Delta_\mathbf{E} = \mathbf{E}_t - \mathbf{E}$
$\Delta_\mathbf{F} = \mathbf{F}_t - \mathbf{F}$
$\mathbf{D} = \mathbf{D}_t$
$\mathbf{E} = \mathbf{E}_t$
$\mathbf{F} = \mathbf{F}_t$
Given $\Sigma_\mathbf{D}$, $\Sigma_\mathbf{E}$ and $\Sigma_\mathbf{F}$ (sensitivity thresholds for distances, travel times and service times, respectively)
if $(\exists (i,j) \in \mathcal{N}^2, \Delta_\mathbf{D} \geq \Sigma_\mathbf{D}$ or $\Delta_\mathbf{E} \geq \Sigma_\mathbf{E}$ or $\Delta_\mathbf{F} \geq \Sigma_\mathbf{F})$ or $L_A \neq \emptyset$ **then**
  Start the real-time VRPTW sub-phase
end if

DVRPTW sub-phase

while $t_{ex} \leq t_{max}$ **do**
  Given $M^{(d)}$, $N^{(d)}$, $\mathcal{F}'^{(d)}$, $\mathbf{D}^{(d)}$, $\mathbf{E}^{(d)}$, $\mathbf{F}^{(d)}$
  Solve the problem (7.4) - (6.12)
end while
Compute and output new routes $\tilde{\mathcal{F}}'^{(d)}$ using matheuristics
Restart the event processing sub-phase

event processing sub-phase is executed that receives information on the disturbances occurring during the deliveries. This procedure evaluates the occurring events and the related effects on the deliveries, i.e., undesired variations in travel time, service time, route distance, and arrival time of TUs. Then, if necessary, the *real-time VRPTW sub-phase* is executed and DVRPTW problems are defined depending on the type of event and affected routes. The real-time solution of the DVRPTW problems is computed according to the formulation described in Section 7.3. Differently from the static phase, the dynamic one is executed in real-time, thus the computational time is critical and must be minimized. It is highlighted that the proposed method considers an event-driven approach, i.e., the event processing sub-phase is activated only when a disturbance occurs. The event-driven approach is particularly effective in this context for two main reasons. First, with periodic updates, there can be a long delay between the happening of an event and the corresponding system's reaction, which is detrimental to the solutions' quality, as shown in [25]. Second, the available technology allows for quick reactions to changes in the environment, with notably GPS tracking and real-time traffic monitoring APIs [26]. Therefore, an event-driven approach is realistically implementable.

The disturbances that trigger the real-time re-routing of the TUs are as follows:

- changes in the travel times matrix $\mathbf{E}$. These variations represent changes in expected travel times due to unforeseen perturbations, such as road traffic. Indeed, unexpected events can cause the delivery to take significantly longer than expected;
- changes in the distances matrix $\mathbf{D}$. The distance matrix contains the distance values for the shortest path between two given addresses. If this shortest path were to become unavailable (e.g., if it is blocked due to an accident or a local event), then it would be replaced by another path, with a different distance value in the matrix;
- changes in vehicle arrivals at any client. While it is not a disturbance per se, the arrival of one of the TUs at any client is considered a significant event. Indeed, it is considered the stopping of a TU for a delivery to be a propitious moment for triggering a re-routing.

Event processing sub-phase: the pseudocode of this subphase is shown in the first part of the following **Pseudocode - Dynamic phase**. The life cycle of an event is as follows: at a given instant

t, the distances matrix $\mathbf{D}_t$, the travel times matrix $\mathbf{E}_t$, the service times matrix $\mathbf{F}_t$ and the list of new vehicle arrivals L_A are collected. This update is considered to be triggered automatically by the provider of the matrices (for example, a routing API). Then, the difference between the online retrieved matrices, i.e., $\mathbf{D}_t$, $\mathbf{E}_t$, $\mathbf{F}_t$, and the offline computed matrices, i.e., $\mathbf{D}$, $\mathbf{E}$ and $\mathbf{F}$, is computed and denoted by $\Delta_{\mathbf{D}}$, $\Delta_{\mathbf{E}}$ and $\Delta_{\mathbf{F}}$ respectively. If any element of these matrices is greater than the given sensitivity thresholds for distances, travel times, and service times (Σ_D, Σ_E and Σ_F respectively), or if L_A is not empty, then the **Real-time VRPTW sub-phase**, described in the second part of the **Pseudocode - Dynamic phase**, is triggered.

DVRPTW sub-phase: the purpose of this sub-phase is to compute new solutions rapidly by minimizing the size of the problem solved at each iteration. This reduces the computation times and increases the system's reactivity. The recalculation problem refers to a partial, static VRPTW containing all the necessary information for the system to react to the considered events. Formally, if it is denoted by d the index of an optimization step, and with $\mathrm{M}^{(d)}$ and $\mathrm{N}^{(d)}$ respectively the number of vehicles and clients affected by the collected events, then solving a recalculation problem is equivalent to solving a VRPTW where the set of TUs of size M is replaced by a smaller set of vehicles of size $\mathrm{M}^{(d)} \leq$M, and the set of clients of size N is replaced by a smaller set of clients of size $\mathrm{N}^{(d)} \leq$N. The matrices collecting the problem's data (i.e., distances matrix, travel times matrix, and service times matrix) are also reduced in size. By analogy, using the notations used in the static phase, it is also defined $\mathcal{F}'^{(d)}$ as a subset of the corresponding set defined in Table 7.1.

It is important to note that index (d) differs from the subscript t used in the event processing sub-phase: the former is used to denote elements relevant to a recalculation problem, whereas the latter simply denotes raw, newly collected matrices. Typically, $\mathbf{D}^{(d)}$ is expected to be of smaller dimension than $\mathbf{D}_t$, and the latter is used as the new reference $\mathbf{D}$ when the event processing sub-phase is recalled.

The recalculation problem, as described above, is then provided to the solver, which generates the relevant new route plans. The solution of the problem is stopped if the optimal solution is found or the execution time t_{ex} exceeds $t_{\max}$, meaning that the optimization period is over.

To formulate a recalculation problem, for each occurring event the algorithm identifies the corresponding affected routes. Four cases can then occur:

(1) the travel time $\mathrm{e}_{i,j}$ between two consecutive clients changes significantly with respect to the offline planning, i.e, $\Delta_{\mathbf{E}} \geq \Sigma_{\mathbf{E}}$ for a couple of clients i and j. A recalculation problem is then defined that includes the two clients and the k-th vehicle that servers the clients, as well as the relevant values for distances, travel times, and service times;

(2) the distance $\mathrm{d}_{i,j}$ between two consecutive clients changes significantly with respect to the offline planning, i.e, if $\Delta_{\mathbf{D}} \geq \Sigma_{\mathbf{D}}$ for a couple of clients i and j, it means that the corresponding distance $\mathrm{d}_{i,j}$ changes significantly with respect to the offline planning. Thus, similarly to case (1) a recalculation problem is then computed;

(3) the service time f_i of a client i changes significantly with respect to the offline planning, i.e, if $\Delta_{\mathbf{F}} \geq \Sigma_{\mathbf{F}}$ for a client i. Thus, similarly to case (1) a recalculation problem is computed;

(4) a TU arrives at a client. A recalculation is then triggered in order to determine if that route can be re-optimised. The associated recalculation problem contains all the information about the vehicle's route: the list of clients that are assigned to this TU is included, as well as the relevant values for distances, travel times, and service times.

7.3 DVRPTW Formulation

This Section presents the formulation of the DVRPTW defined and solved in the DVRPTW sub-phase of the dynamic phase of the algorithm described in Sub-section 7.2.2. As introduced in Sub-section 7.2.2, the mathematical model solved in the DVRPTW sub-phase of the dynamic phase, differs from the 3L-CVRPTW sub-phase of the static phase, presented in Sub-section 7.2.1. In particular, in this case, only the constraints of the VRPTW are considered, while removing the ones related to the capacity limits of the vehicle. In fact, the static phase ensures that the cargo assigned to each TU respects its volume and the maximum supported weight limits. Moreover, each client is associated with at most one TU, and the bins are loaded inside the TUs according to the shipment plan. Thus, the DVRPTW formulation is defined for each TU, whose route is affected by a disturbance. The parameters and variables of the problem are reported in Tables 7.2 and 7.3 and the problem is formulated as follows:

$$\min(\mathrm{h}_1 \sum_{k=1}^{\mathrm{M}^{(d)}} \sum_{j=1}^{\mathrm{N}^{(d)}} \mathrm{C}^k t^k_{0,j} + \mathrm{h}_2 \sum_{k=1}^{\mathrm{M}^{(d)}} \sum_{i=1}^{\mathrm{N}^{(d)}} \sum_{j=1}^{\mathrm{N}^{(d)}} \mathrm{V}^k \mathrm{d}_{i,j}\ t^k_{i,j} + \mathrm{h}_3 \sum_{k=1}^{\mathrm{M}^{(d)}} \sum_{j=1}^{\mathrm{N}^{(d)}} \delta^k_j) \tag{7.4}$$

subject to: (7.5)

$$\sum_{i=0}^{\mathrm{N}^{(d)}} t_{i,l} - \sum_{j=1}^{(\mathrm{N}^{(d)}+1)} t_{l,j} = 0, \forall l \in \mathcal{A}^{(d)}, i \neq j \tag{7.6}$$

$$\sum_{i=1}^{\mathrm{N}^{(d)}} t_{i,l} = 1, \forall l \in \mathcal{A}^{(d)}, l \neq i \tag{7.7}$$

$$\sum_{j=1}^{\mathrm{N}^{(d)}} t_{0,j} = 1 \tag{7.8}$$

$$\sum_{i=1}^{\mathrm{N}^{(d)}} t_{i,(\mathrm{N}^{(d)}+1)} = 1 \tag{7.9}$$

$$\delta_i \geq \mathrm{A}_i - s_i, \forall i \in \mathcal{A}^{(d)} \tag{7.10}$$

$$s_i \leq \mathrm{Z}_i - \mathrm{f}_i,\ \forall i, j \in \mathcal{A}^{(d)} \tag{7.11}$$

$$s_i + \mathrm{f}_i + \mathrm{e}_{i,j}\ t_{i,j} - s_j + \mathrm{U} t_{i,j} \leq \mathrm{U},\ \forall i, j \in \mathcal{A}^{(d)} \tag{7.12}$$

$$s_i + \mathrm{f}_i + \mathrm{e}_{i,j}\ t_{i,(\mathrm{N}^{(d)}+1)} - \mathrm{Z}_{\mathrm{N}^{(d)}+1} + \mathrm{U} t_{i,j} \leq \mathrm{U}, \forall i, j \in \mathcal{A}^{(d)} \tag{7.13}$$

where $\mathrm{U} = max_{i,j}\ \{\mathrm{Z}_i - \mathrm{A}_i + \mathrm{e}_{i,j}\}$.

The objective is to minimize the total travel costs and the time windows violations. The firsts are computed as the summation, for all the TUs, of the base travel costs C^k (i.e., all the costs of the k-th TU which are not related to the travelled distance and could include rental fees, maintenance, etc.) and the variable travel costs V^k (i.e., the costs of the k-th TU related with distance, such as gasoline consumption). The latter instead are computed as penalties proportional to the advance/delay time of the k-th TU at j-th client with respect to the corresponding opening time. In the objective function, the corresponding three terms are weighted, according to the needs of the application in which the model is executed, using three parameters h_1, h_2, and h_3 whose sum is equal to 1. The company is then in charge of deciding the weights of the terms. Constraints (7.6)-(7.9) are related to the feasibility of the routes: (7.6) imposes that the sum

of the routes entering a client l must be equal to the sum of the routes exiting the same client and prohibits both the creation of loops on the same client i and in the network (i.e., the TU k can only go from i to j or from j to i); (7.7) imposes that each client must be visited only once. Constraints (7.8) and (7.9) are related to the starting and ending clients of each route and ensure that each TU must start its route from the initial depot (indicated by index 0) and end it at the final depot (indicated by index $N^{(d)}+1$, with the convention that this final depot is the same for each optimization step d). The last constraints (7.10)-(7.13) are related to the time windows, which are defined as an interval of values $[A_i,Z_i]$ where A_i is the opening time for delivery of client i and Z_i is the closing time. Constraints (7.10) impose that the penalty δ_i^k is greater than 0 if the TU k arrives before clients' i opening time. Constraints (7.11) are hard constraints and impose that each TU must arrive at each related client before its closing time, while (7.12) and (7.13) impose that the arrival time of each TU k to each client j must be higher than or equal to the sum of the arrival time at client i, the service time at client i and the travel time from client i to client j. An example of routes assignment is presented in Fig. 7.5, where two routes are assigned respectively to two transport units TU1 and TU2. The five clients and two depots are represented by labelled nodes which are connected by directed arcs. Each client i visited by TU k is characterised by its time window $[A_i,Z_i]$ and service time f_i^k. The weight of the generic arc i,j is characterised by a couple $(d_{i,j}, e_{i,j})$ representing respectively its length and travel time. The base and variable costs of each TU are represented by pairs (C^k, V^k) with k={1,2}. Both routes start and end in the initial and final depots, the route assigned to TU1 includes the clients {2,5} while the route assigned to TU2 includes the clients {1,3,4}.

Table 7.3: Parameters and variables of the DVRPTW

Name	Description
Input Parameters	
$M^{(d)}$	Number of TUs affected by a disturbance
$N^{(d)}$	Number of clients affected by a disturbance
$d_{i,j}$	Kilometers from client i to j
$e_{i,j}$	Travel time from client i to j
f_i	Service time at client i
A_i	Opening time of time window of client i
Z_i	Closing time of time window of client i
U	Upper bound for arrival time at each client
Output Variables	
d	Index of the optimization step
i,j,l	Indices of clients
$t_{i,j}$	Binary variable indicating whether the TU runs from client i to client j (1) or not (0)
$t_{0,j}$	Binary variable indicating whether TU runs from the initial depot to client j (1) or not (0)
$t_{i,(N^{(d)}+1)}$	Binary variable indicating whether the TU runs from client i to the final depot (1) not (0)
s_i	Arrival time of the TU at client j
δ_i	Advance/delay of the TU arrival at client j with respect to opening time A_j

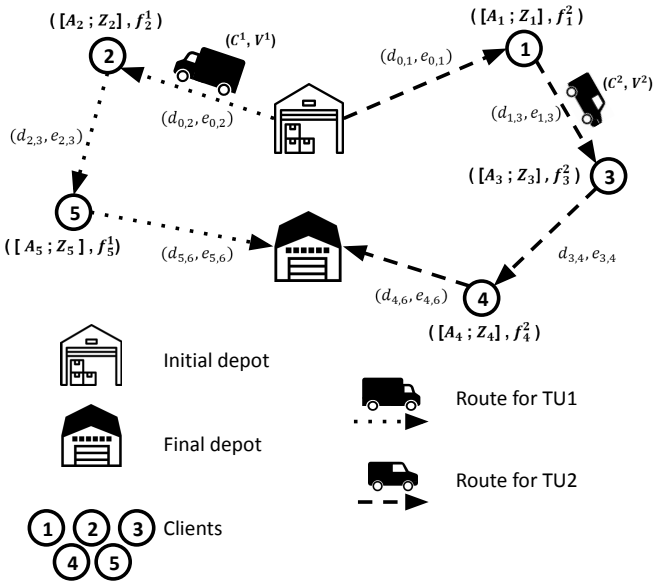


Figure 7.5: Example of routes assignment for two TUs, based on the solution of the proposed CVRPTW.

7.4 Heuristics-based Solution for the DVRPTW

To implement the TUs' re-routing in the dynamic phase of the algorithm that solves (7.4) - (6.12), a heuristic-based approach based on a genetic algorithm (GA) is defined [27]. In the following, firstly it is defined the chromosome model for the DVRPTW as well as the steps of the procedure and how the GA is applied. GAs are optimization algorithms that draw inspiration from the natural principles of the evolution of the species and the survival of the fittest. GAs work iteratively on a population of candidate solutions to the problem. The fitness, i.e., the value of the objective function associated with each solution in the population, rules the iterative selection schema so that solutions with high fitness have a high likelihood to mate and form the offspring population. Once the offspring population is selected, special operators emulating genetic crossover and mutation are randomly applied to create new solutions. These steps are then iterated in order to obtain new solutions with increased fitness until a pre-specified stopping criterion is met. Although GAs were originally designed to explore large multi-dimensional search hyper-cubes, a variety of different versions of these algorithms is available in the literature to handle special cases as multi-modal and constrained problems, as the one considered in this research ([27],[28]).

7.4.1 Choromosome Model and Fitness Function

A. Chromosome model for DVRPTW. In GAs the representation of a solution is very important because it defines the search space of the solution and the operators that can be used to explore it. This, in turn, influences algorithm complexity and convergence. In this case, a chromosome represents a list of routes, each route being a sequence of clients. Moreover, the following metadata are attributed to each route within the chromosomes:

- a TU ID representing which vehicle is assigned to the considered route;

Pseudocode - Initial population construction

Initialization

α = Previous solution
Φ = Desired population size
S = Maximum number of shuffling attempts
q = Empty list /*q is the output population of solutions*/
M = Number of routes in s

Algorithm

```
for n = 1:Φ do
    α_n = α
    for k = 0:(M−1) do
        r_k = Route k in α_n
        u_k = Removed clients in r_k
        if u_k ≠ Empty list then
            r_k = Reinsert clients (r_k, u_k)
        end if
        a = 0 /*a is a counter*/
        while r_k is not valid do
            if a < S then
                r_k = Shuffle movable clients (r_k)
                a = a + 1
            else
                r_k = Remove client (r_k)
                a = 0
            end if
        end while
    end for
    /*α_n now contains valid shuffled routes*/
    q = Add individual (q, α_n)
end for
return q
```

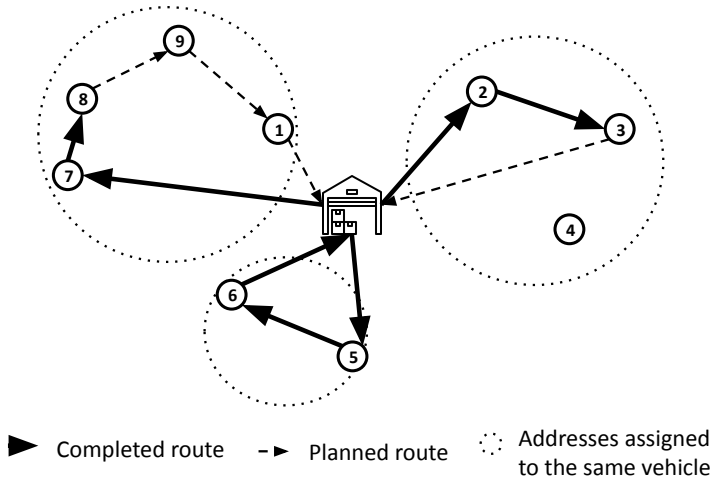
- the index of the last visited client in the route;
- a list of removed clients, containing all the clients that were assigned to the vehicle, but cannot be served anymore due to dynamic changes.

The purpose of this approach is to have both the routing information and data required to account for the dynamic nature of the problem: the index of the last visited client lets the algorithm establish which segments of the route plans are mutable, and the removed clients are retained for eventually reinserting them later.

An example of a route plan and its chromosome model are illustrated in Fig. 7.6. In the upper part of the illustration, clients are represented by circles including the corresponding index i with $i \in \{1, 2, \ldots, 9\}$. In particular, the initial and final depots are coincident. Three TUs denoted as TU1, TU2, and TU3 are dispatched and are all following their assigned route plan, represented by arrows. Bold arrows represent the parts of the routes already completed by the TUs, while dashed arrows represent the routes' portions that still have to be executed. Below the graph, a representation of the associated chromosome model is reported. In particular, vehicle TU1 must start with the initial depot, then visit clients 2, and 3, and come back to the depot in the specified order. In the considered time instant, the TU has already visited clients 2 and 3, while the current client is 2 (light gray colored in the chromosome). Also, client 4 is assigned to vehicle TU1, but it is deemed impossible to serve and is therefore retained in a list of removed clients (dark grey colored in the chromosome). The detailed information is then stored in a route model identified by vehicle TU1's ID. The routes corresponding to vehicles TU2 and TU3 are modeled in the same way.

Fitness function for GA-based VRPTW: the fitness of a chromosome is a function evaluating its quality. The GA's purpose is to improve this indicator. In this case, the fitness function is given directly by the objective function (7.4), which the algorithm aims to minimize.

(a) Route plan example



(b) Chromosome model corresponding to the route plan

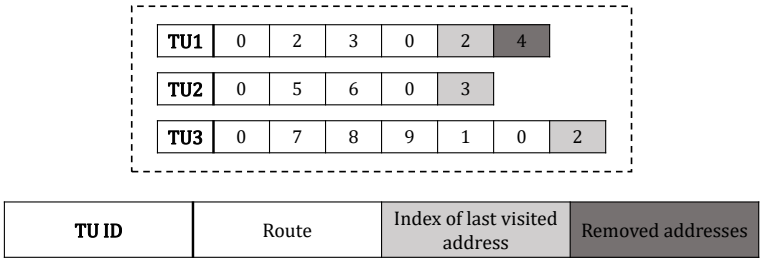


Figure 7.6: Example chromosome modelisation

7.4.2 Initial Population and Operators

Initial population generation: GAs cannot construct solutions from scratch, thus a method to generate new solutions from existing ones is required. This method must favor diversity and hence reduce the probability of reaching a local optimum too rapidly.

The procedure to build an initial population is described in **Pseudocode - Initial population construction**. In particular, the bases used to create new solutions are either the route plans established during the static phase (for the initial re-routing step) or the route plans computed at the previous re-routing step denoted by α. Within this initial solution, the same procedure is applied to each route r_k, with the subscript k referring to the TU associated with that route. First, the clients that were removed previously (if any), denoted by u_k, are reinserted into the route. Then, the clients are shuffled until the resulting solution is found to be valid, or until a certain number of attempts S is reached. If no valid solution is found, a client is removed, and the clients are shuffled again. This way, new valid routes from the existing ones are generated. Once every route is shuffled, a new solution α_n is obtained, where n varies between 1 and the desired population size Φ. This solution is then added to the population, denoted by q.

Crossover operators: crossovers must generate new “children" individuals that retain the qualities of their parents. They happen randomly at each step (or generation) of the GA: any given individual is crossed over with another individual with a given **crossover probability** ρ_χ.

Let p_1 and p_2 be two parent chromosomes. A random route uniformly in p_1 is selected, and its counterpart in p_2. The said counterpart is determined with the route's associated TU ID. With those two routes at disposal, two children's chromosomes are obtained by applying one of the operations below. The operator is chosen randomly, and both have an equal chance to be picked.

- **Route swap crossover**: the route from p_1 is swapped with its counterpart from p_2, and vice-versa for the second child. The other routes in the chromosome are left unchanged.
- **Two-point crossover**: this procedure is a slightly modified version of the partially-mapped crossover described in [28] for the travelling salesman problem. It is applied to the selected routes, and once again the other routes are not modified.

Crossover operators are chosen for their compatibility with this chromosome model. They ensure that assignments remain fixed during the dynamic phase and that no client is visited twice (except potentially for the first and last ones).

Mutation operators: mutations are spontaneous, random changes within a few chromosomes at each generation. Their purpose is to maintain variety in the solutions and widen the search space. At each generation, a number of individuals are chosen randomly to undergo a mutation with a given **mutation probability** ρ_μ. This mutations are conducted analogously to the crossovers described previously: one of the individual's routes is selected uniformly, and applied to it one of the following heuristics, described in [29].

- **Intra-route swap**: two random clients within the route are exchanged.
- **Intra-route shift**: a random client is selected and shifted to another random position in the sequence.

Similarly to crossovers, these operators are chosen because of their adaptability to this model.

Selection process: in order to increase the overall fitness of the individuals, a selection is conducted in the population at each generation. Let us consider a population of I individuals of fitness $F_i > 0$ ($i \in [\![1; I]\!]$), and let P be the desired population size. For the purposes of this chapter, it is considered a simple selection of the fittest: if $P \leq I$, the P individuals with the highest fitness F_i are selected. If however $P > I$, all I individuals are selected, and the individual with the highest fitness is added another $P - I$ times.

7.5 Experimental Results

In this section, the tests conducted with the aim of demonstrating the validity of the proposed method are presented. For the sake of completeness, two different test sets are presented. The first set focuses on the dynamic phase of the algorithm and considers the comparison with the results of the well-known static methodology by Solomon, *et al.* in [30] showing that the dynamic approach can ensure several improvements in real-time scenarios. The second one considers real data of a full workday provided by the Italian logistics company E80 Group [31]. It is highlighted that for this first test, the focus is on the dynamic phase of the proposed methodology due to the lack, in the related literature, of comprehensive works that present approaches for the combined solution of the 3L-CVRPTW and dynamic VRPTW, while an evaluation of the static phase with respect to the literature is discussed in [10].

The static phase of the algorithm is implemented in Matlab2020a [32], which is suitable for the exact solution of complex optimization problems, running on a 2.20 GHz Intel Corei7-8750H CPU with 32 GB RAM. Differently, the dynamic phase is implemented using Kotlin, an object-oriented programming language running on the Java Virtual Machine, which is particularly suitable for the execution of GAs, running on a 3.4 GHz AMD Ryzen 5 2600 CPU with 16 GB RAM. For the implementation of the comparison it han been used Google's OR tools [33], and more specifically its Java API (which is completely inter-operable with Kotlin at no performance cost). It is instantiate a SCIP (Solving Constraint Integer Programs) solver, which is based on branch-and-bound methods and particularly suitable for MILP problems. The information regarding TUs, clients, and bins is modelled using object-oriented programming, while the communication between the two subsystems was made using the JSON (JavaScript Object Notation) format (i.e., a lightweight and widely used readable data format used to represent structured data and data exchange between different systems[34]). For the sake of completeness, it is remarked that the parameters used in the following tests have been defined by practical requirement of the real case study, or chosen according to the results achieved in some preliminary experiments with various parameter values, and in the interest of brevity in the rest of the chapter it is only reported the best-performing parameter values since fine-tuning the GA parameters is not a primary focus. Specifically, for the GA the crossover and mutation probabilities are tested ranging from 0.1 to 1 in increments of 0.1. Regarding the selection process, both selecting the fittest individuals and using roulette wheel selection are explored, with the former yielding slightly better results.

7.5.1 Test of the Dynamic Phase with a Literature Benchmark

In this subsection, it is introduced the metrics and disturbance model used to conduct simulations with the dynamic phase of the algorithm described in Section 7.2.2. Firstly, all the performance indices to evaluate the results of the simulations are listed and described. Then, there is the description of the model used to simulate real-time workdays. In particular, the models presented are for three types of perturbations: travel time variations, service time variations, and spontaneous decisions from the drivers to reorder the clients. For the simulations and for the sake of clarity, it has been decided not to introduce perturbations in the distance matrix $\mathbf{D}$, because its role is similar to the travel times matrix $\mathbf{E}$. Finally, the well-known benchmark by Solomon [30] is used as a basis for the first set of simulations that justifies the system's interest. The goal of this test is threefold: first, it is demonstrated that the dynamic approach is superior to a static approach in terms of solution quality, it is proved that the use of the GA is preferable to the use the MILP model defined in the static phase for the optimatilty of solution ad fast computation time; then, the assumption made are validated by considering a limited number of clients and vehicles is sufficient to improve the route plans impacted by dynamic events.

In order to achieve these objectives, it used a well-known static VRPTW instance as the basis of the simulation and extended it with the proposed dynamic model (described in Section 7.5.1.2) in order to simulate a dynamic problem. The results of the proposed algorithm are compared with the results given by a static approach, that keeps the same route plans during the entire workday without taking dynamic events into account. The route plans used for this static approach are given by the best-known solution [35].

Specifically, the instance tested is the C101 instance of Solomon's benchmark [30], which is widely used in the literature to evaluate the performance of both static and dynamic models. The instance choosen has 100 customers and 10 vehicles, which are relatively high numbers that will let us evaluate the performance of the proposed algorithm in terms of computing time. Moreover, it is expected the best-known solutions to be well-optimized on such a benchmark, meaning that the static approach provides the best results attainable without adapting to dynamic changes.

Since Solomon's instances do not specify travel times, it is considered that travel times and distances are equal in value. Finally, since the proposed dynamic phase requires an initial route plan, it is used the aforementioned known solution for that purpose.

7.5.1.1 Performance Indicators and Disturbance Model

The performance indicators used to evaluate the efficiency of the dynamic phase of the algorithm are as follows:

- **Objective Function at the End of the workday (OFE)**: representing the total routing costs at the end of the working day, thus considering also the re-routing costs. Consequently, the OFE is calculated based on (6.1):

$$\mathrm{OFE} = \mathrm{h}_1 \sum_{k=1}^{M} \sum_{j=1}^{N} \mathrm{C}^k \widetilde{t}^k_{0,j} + \mathrm{h}_2 \sum_{k=1}^{M} \sum_{i=1}^{N} \sum_{j=1}^{N} \mathrm{V}^k \widetilde{\mathrm{d}}_{i,j}\ \widetilde{t}^k_{i,j} + \mathrm{h}_3 \sum_{k=1}^{M} \sum_{j=1}^{N} \widetilde{\delta}^k_j \tag{7.14}$$

  where $\widetilde{D}_{i,j}$ are the actual distances, $\widetilde{t}^k_{i,j}$ are the actual paths assigned to the vehicles, and $\widetilde{\delta}^k_j$ are the actual arrival times gathered at the end of the workday, instead of the values computed offline.

- **Objective Function on Average (OFA)**: this is the average value obtained for the objective function (7.4) when treating a recalculation problem (defined in 7.2.2). While OFE is a performance indicator for the overall solution, OFA shows the capacity of the system to respond to dynamic events. It is calculated by averaging all the objective function values computed at each optimization step d with (7.4).

- **Total Travelled Distance (TTD)**: representing the total distance traveled by all TUs over the entire workday. Using the notation introduced in Section 7.2.1, it is given by:

$$\mathrm{TTD} = \sum_{k=1}^{M} \sum_{i=1}^{N} \sum_{j=1}^{N} \mathrm{d}_{i,j}\ t^k_{i,j}. \tag{7.15}$$

- **Total Travel Time (TTT)**: similarly to TTD, it is the total travel time of all TUs during the workday. This also includes service times. Using the notation introduced in Section 7.2.1, this index is formulated as:

$$\mathrm{TTT} = \sum_{k=1}^{M} \sum_{i=1}^{N} \sum_{j=1}^{N} \mathrm{e}_{i,j}\ t^k_{i,j}. \tag{7.16}$$

- **Average Computational Time (ACT):** this indicator is obtained by averaging the time to compute each recalculation problem expressed in seconds.

- **Average number of Vehicles on Recalculation (AVR)**: it represents the average number of vehicles included in the recalculation problems at each disturbance occurrence. It allows for determining how much the problem size is reduced during the dynamic optimization process.

- **Average number of Clients on Recalculation (ACR)**: it represents the average number of clients included in the recalculation problems at each disturbance occurrence.

- **Relative Standard Deviation (RSD)**: in statistics, it is defined as the ratio of the standard deviation to the mean of a distribution. It is used as a measure of dispersion quantify the stability of the other indicators defined above.

To evaluate the performance of the dynamic phase and show the improvements induced by the proposed algorithm in case of disturbances, the performance obtainable with the only application of the static phase and the ones obtainable with the whole algorithm considering the benchmark are compared.

7.5.1.2 Disturbance Models

In order to retrieve the parameters needed for the simulation of the entire workday, the three most frequent disturbance scenarios are defined and represented by three corresponding models.

A. Travel time variations model: similarly to what is proposed in [25], it is accounted for commuting traffic by associating a coefficient to travel times for each period of the day (respectively m_{TT}, l_{TT} and a_{TT} for the morning, lunchtime and afternoon), and added a randomly-generated perturbation. This perturbation follows a normal distribution of mean $\mu_{TT} = 0$, and of standard deviation σ_{TT}. Negative values are brought back to zero to represent the absence of perturbation.

B. Service times variation model: it takes into account the changes in the service times and adds a penalty depending on the number of clients changed between the pre-planned route and the effective dynamic route. This penalty is introduced because, when the MILP solves the static 3L-CVRPTW problem, goods are loaded in trucks with a Last-In-First-Out (LIFO) assumption. Therefore, if the dynamic solver changes the order in which the clients are served, goods become harder to unload. This penalty is modeled by a linear function: if x is the number of re-ordered clients in a route for vehicle k (compared to the offline planning for the same route), then the service time for client i will become $f_i^k + x\pi_{ST}$, where π_{ST} is a penalty coefficient.

C. Clients' reordering by the driver: probability for any vehicle to visit a client different from the one that the route plan suggests. This client is chosen with a uniform probability among the clients that have not been visited, excluding the first and last clients. This probability, denoted by ρ_{OC}, models the fact that users of a re-routing algorithm can decide not to follow the received instructions or might make mistakes leading to wrong destinations. Whenever a vehicle arrives at a client, a Bernoulli trial of parameter ρ_{OC} is conducted to determine whether such changes occur. All the parameters used in the simulations are listed and described in Table 7.4.

7.5.1.3 Evaluation with a Benchmark Instance

In the following series of tests, it is used the parameters listed in Table 7.5 to generate dynamic events. The objective function is weighted giving using the three terms, (e.g. the base cost and its weight h_1, the variable cost and its weight h_2 and the delivery delay and its term h_3), the same importance. The other values were chosen by analogy with the realistic scenarios presented in section 7.5.2. The same test is run four times with different sensitivity thresholds Σ, ranging between 0 and 3. The numerical results are shown in Table 7.6.

It is highlighted that the average computation times are not provided for the reference tests (without dynamic improvement) because the latter directly use the routes plans provided statically and therefore do not require computation during the simulation. In addition, there is the plot of the evolution of the recalculation problems' size ACR and of the computation time ACT with respect to the sensitivity threshold (Fig. 7.7).

In order to prove the effectiveness of the proposed metaheuristics-based approach over an exact approach for the dynamic phase, an exact solver for the dynamic phase has been implemented. All the variables defined in table 7.3, the constraints (7.6)-(6.12) and the objective function are

Table 7.4: Simulation parameters description

Variable	Description
General parameters	
η	Number of times the simulation instance is run.
τ_{call}	Time interval (in seconds) between two requests to the external data source.
τ_{start}	Time of the day (in seconds) at which the delivery starts.
τ_{ML}	Time of the day (in seconds) considered to be the limit between morning and lunchtime.
τ_{LA}	Time of the day (in seconds) considered to be the limit between lunchtime and afternoon.
ρ_{OC}	Probability for a vehicle to visit a client in a different order than the one instructed by the system.
Σ	Sensitivity threshold.
Parameters of the Genetic Algorithm	
ψ	Desired number of individuals in the population.
ν	Number of generations, i.e., iterations of the GA.
ρ_{χ}	Crossover probability
ρ_{μ}	Mutation probability
Parameters of the perturbation models	
m_{TT}	Coefficient by which travel time values are multiplied in the morning.
l_{TT}	Coefficient by which travel times values are multiplied during lunchtime.
a_{TT}	Coefficient by which travel times values are multiplied in the afternoon.
μ_{TT}	Mean travel time perturbation in the normal distribution.
σ_{TT}	Standard deviation of the travel time perturbation in the normal distribution.
π_{ST}	Penalty (in seconds) added to service times for each change in the planned route.

implemented using the aforementioned library. The results are reported in Table 7.6. The first observation from the results regards the significant improvement in solution quality induced by the dynamic phase of the algorithm over the reference test: the OFE index in Table 7.5, regarding the value of the objective function computed at the end of the workday, is improved on average by 4.6%. This shows the effectiveness of the proposed algorithm when perturbations are introduced in a static VRP. Instead, the second observation regards the different sensitivity thresholds tested, the solutions vary very little in quality. For example, the relative standard deviation of the OFE indicator in Table 7.5 is only 0.17% with dynamic improvement. As shown in Fig. 7.7, where results are plotted for the dynamic phase of the algorithm, it can be related with the average recalculation problem size, indicated by AVR and ACR (respectively number of vehicles and clients). With a sensitivity threshold of 3, both indicators are reduced significantly: the average number of vehicles in each recalculation, which was stable at around 10, drops to 3.4; and the average number of clients goes from 100 to 40.6. This leads to a notable reduction in computation time at each recalculation, with the indicator of the average computation time ACT

Table 7.5: Simulation parameters - instance C101

General		Genetic Algorithm		Perturbation	
Parameter	**Value**	**Parameter**	**Value**	**Parameter**	**Value**
η	5	ψ	100	m_{TT}	1.20
τ_{call}	5 [s]	ν	100	l_{TT}	0.50
τ_{start}	0 [s]	ρ_{χ}	0.80	a_{TT}	1.20
τ_{ML}	674 [s]	ρ_{μ}	0.30	μ_{TT}	0
τ_{LA}	787 [s]			σ_{TT}	0.80
ρ_{OC}	0.05			π_{TT}	1
Σ	$\in [0, 1, 2, 3]$				
h_1	0.33				
h_2	0.33				
h_3	0.33				

Table 7.6: Simulation results on instance C101 of Solomon's benchmark

Σ	**OFE**	**OFA**	**TDT**	**TTT**	**AVR**	**ACR**	**ACT**
With exact-solver-based dynamic improvement							
0	0.4286	0.4198	689.4	795.2	10	100	3780,18
1	0.4278	0.4289	662.1	766.5	10	100	3720,42
2	0.4279	0.4281	664.3	764.3	9.0	94.0	3300,03
3	0.4284	0.4398	683.8	787.6	3.2	40.1	900,74
With GA-based dynamic improvement							
0	0.4302	0.4210	692.5	797.2	10.0	100.0	6,08
1	0,4287	0,4292	663.2	767.6	10.0	100.0	6,16
2	0.4286	0.4289	666.6	765.0	9.0	94.0	6,59
3	0.4288	0.4415	684.0	788.4	3.4	40.6	2,31
Without dynamic improvement							
0	0.4501	0.4503	1074.7	1249.2	10.0	100.0	-
1	0.4504	0.4503	1082.3	1256.6	10.0	100.0	-
2	0.4507	0.4499	1076.7	1252.3	9.0	92.3	-
3	0.4474	0.4727	1021.4	1185.2	2.2	29.0	-

going down from 6598.6 ms to 2315.0 ms (see Table 7.6). In other words, it has been confirmed the assumption that it is not necessary to recalculate the entire VRP whenever any change in the parameters is noticed by the system: it is enough to consider a part of the problem that was significantly affected by external events. Moreover, a good trade-off between computation time and solution quality can be found by tuning the system's sensitivity to those events. Let us now compare the results obtained by the GA-based solver, with those obtained by the exact solver. The first thing to note is that as it could be expected, the OFE, OFA, TDT and TTT tend to be better with the exact solver. However, the improvement is not very significant (less than 1% on average). On the other hand, the average computing time, is much higher for the exact solver, with some recalculations taking up to an hour or more. Even with higher sensitivity thresholds (i.e. less clients and TUs to manage on average), computing exact solutions takes several minutes on average. This is reflected by the high number of variables that is equal 1206 for Σ=0,1 , 1128 for Σ=2 and 482 for Σ=3. Besides, computing time increases very fast with this number of variables, meaning that an exact solver would not scale well on larger instances (which would be common in real-life applications).

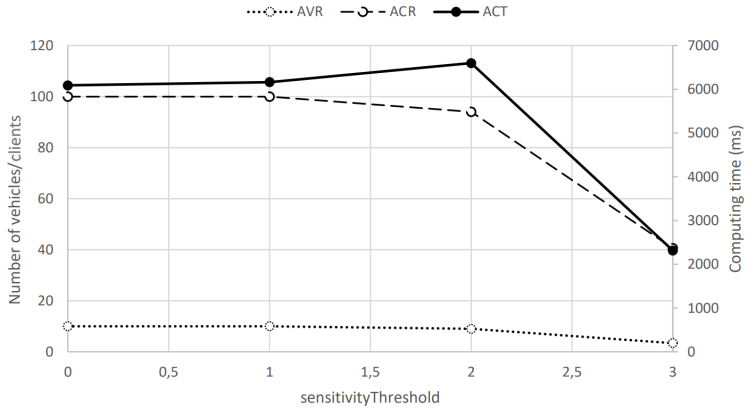


Figure 7.7: Simulation on instance C101 of Solomon's benchmark - Influence of the Σ parameter on recalculation problems

On a practical standpoint, it can be noted that computing times for this large instance are between 2 and 6 seconds with the proposed GA-based approach which confirms that it could be used in a real-world application since drivers could reasonably wait for the system to compute route plans. It has also been shown that, because of the high average computing time for minimal performance gain, exact solvers are not fit for the dynamical recalculation of route plans: it would not be realistic for drivers to have to wait for the computation every time an event occurs.

7.5.2 Test of the Proposed Matheuristics Algorithm With a Real Case Study

In this subsection are described the results obtained by applying the proposed algorithm to a real dataset provided by the Italian logistic company E80 Group. In particular, this dataset is composed of 52 clients distributed all over Italy, 299 bins with different heights and weights but the same base area (they lie on the EU standard pallet of dimension 120x80 cm), and 2 different types of TUs available with a virtually infinite fleet and whose parameters are reported in Table 7.7.

Table 7.7: Parameters of the real case study

TU	L [mm]	W [mm]	H [mm]	Q [kg]	C	V
1	800	244	260	10000	350	600
2	1360	244	260	24000	350	600

Firstly the results obtained as the output of the static phase are illustrated, and then also the results of the dynamic phase when the planned routes are affected by (1) drivers' decisions impacting both the visit order of the clients and the path used to reach two clients, thus changing the distance matrix and (2) disturbances that provoke substantial changes in travel times.

7.5.2.1 Test of the Static Phase

In order to fasten the execution of the static phase, the VRP and the CLP are executed in parallel, respectively referring to the addresses that contain anomalies (see Section 7.2.1), clusters, and TUs, using the Parallelization Toolbox provided by Matlab. This is possible because each loop

executes separately the relative mathematical models for different instances, and so each iteration is completely independent from the other.

The computation time for the whole static phase is equal to 3896.1 seconds and assigns the bins of the clients to 16 TUs, obtaining then 16 routes whose total distance to be traveled is equal to 8012 km distributed all over Italy. The average fill ratio in terms of supported weight (i.e., the percentage of weight carried by the TU related to its total supported weight) is equal to 78.71%, while the average fill ratio in terms of volume (i.e., the percentage of the volume occupied by the bins related to the total TU's volume) is equal to 47.65%. This is motivated by the fact that the bins of the available dataset present a high weight in a limited volume and so in many cases the total transported weight reaches 80%-90% of the total supported weight with a limited number of bins.

7.5.2.2 Test of the Dynamic Phase in Case of Unexpected Human Behaviour

A series of tests aimed at studying the dynamic algorithm's performance is conducted, with respect to unexpected decisions (or mistakes) made by the drivers, e.g., arrival at a client earlier than planned.

Firstly, a simulation instance considering a variation in the probability for a driver to deviate from the offline planned route (ρ_{OC}) is defined. The values of the parameters used for these simulations are listed in Table 7.9. To account for the random nature of this simulation, the test are run three times for each value of ρ_{OC}, and report the average results. These results are presented in Table 7.8, and the evolution of the OFE index with respect to ρ_{OC} is plotted in Fig. 7.8. As means of comparison, a test is run in the same instance, but without using the dynamic phase. In this case, when a driver deviates from the offline planned route, the client visited is brought earlier in the route to account for that decision, but no other changes are made. The results obtained with this reference system are also reported in Table 7.8, and the evolution of OFE is represented in Fig. 7.8.

Table 7.8: Influence of perturbations induced by human decisions on real case study

	With dynamic improvement				**Without dynamic improvement**			
ρ_{OC}	**OFE**	**OFA**	**TDT**	**TTT**	**OFE**	**OFA**	**TDT**	**TTT**
0	0.4605	0.4452	9729478	279764	0.464737	0.448715	10039285	288581
0.1	0.4650	0.4477	10060509	288427	0.470966	0.454864	10503437	300347
0.2	0.4664	0.4484	10167027	290338	0.470915	0.457515	10499606	298766
0.3	0.4661	0.4480	10144884	288918	0.47201	0.452939	10581203	301620
0.4	0.4681	0.4477	10292073	293190	0.481345	0.466359	11276805	483889
0.5	0.4716	0.4545	10551679	299514	0.478708	0.465811	11080317	632167
0.6	0.4699	0.4510	10424107	297125	0.484563	0.470222	11516585	651496
0.7	0.4709	0.4525	10503468	299753	0.476959	0.459637	10950006	310439
0.8	0.4682	0.4489	10300340	294173	0.483042	0.470069	11403218	491570
0.9	0.4661	0.4493	10140691	289462	0.480251	0.465399	11195241	319867
RSD	0.7	0.6	2.4	2.1	1.3	1.6	4.4	35.5

The obtained results highlight first that the solution quality is improved by the dynamic phase of the algorithm. In particular, overall tests, there is on average a 1.89% improvement on the OFE index (i.e., objective function at the end of the workday), 2.54% on the OFA index (i.e., the objective function on average), and 6.08% on the TTD index (i.e., the total travelled

Table 7.9: Simulation parameters - real case study with human decision

General		Genetic Algorithm		Perturbation	
Parameter	**Value**	**Parameter**	**Value**	**Parameter**	**Value**
η	3	ψ	100	m_{TT}	1.25
τ_{call}	300	ν	100	l_{TT}	0.50
τ_{start}	32400	ρ_χ	0.80	a_{TT}	1.25
τ_{ML}	43200	ρ_μ	0.30	μ_{TT}	0
τ_{LA}	50400			σ_{TT}	300
ρ_{OC}	$\in [0.0, 0.9]$			π_{TT}	600
Σ	1000				
g_1	0.50				
g_2	0.50				
h_3	0.33				
h_2	0.33				
h_3	0.33				

distance) with respect to the results obtained without the dynamic rerouting. Figure 7.8 shows the considerable improvement of the objective function recomputed after the re-routing. Intuitively, this improvement tends to be more significant for higher values of ρ_{OC}, which confirms that the proposed algorithm is adaptable to unplanned changes. The second point of interest is that the stability of the solutions is also improved with respect to the static case, as shown by the lower Relative Standard Deviation (RSD) values reported in Table 7.8 for the dynamic routing case.

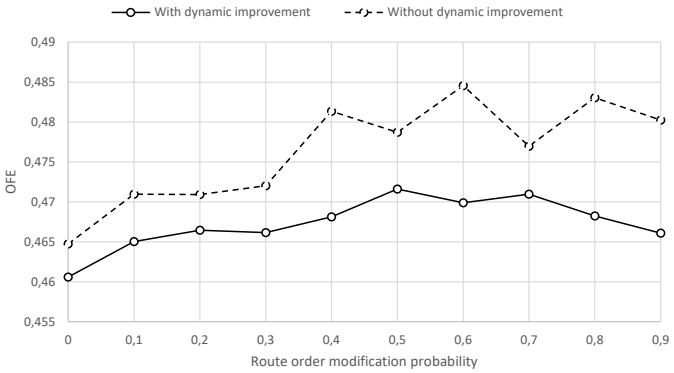


Figure 7.8: Influence of perturbations induced by human decisions - Objective function at the end of the day (OFE)

It is then possible to deduce that the dynamic phase of the algorithm is able to optimize resource usage in spite of unexpected perturbations induced by human decisions and mistakes.

7.5.2.3 Test of the Dynamic Phase in Case of Disturbances Affecting Travel Times

This last series of tests evaluates the performance of the proposed algorithm in case of high variation in travel times during the workday. To this aim, six tests are executed. The corresponding parameters are listed in Table 7.10, and variable values for the standard deviation of the normal

Table 7.10: Simulation parameters - real case study with travel time changes

General		Genetic Algorithm		Perturbation	
Variable	**Value**	**Variable**	**Value**	**Variable**	**Value**
η	5	ψ	100	m_{TT}	1.25
τ_{call}	60	ν	100	l_{TT}	0.50
τ_{start}	32400	ρ_{χ}	0.80	a_{TT}	1.25
τ_{ML}	43200	ρ_{μ}	0.30	μ_{TT}	0
τ_{LA}	50400			σ_{TT}	$\in [0, 1500]$
ρ_{OC}	0.10			π_{TT}	600
Σ	1000				
g_1	0.50				
g_2	0.50				
h_1	0.33				
h_2	0.33				
h_3	0.33				

Table 7.11: Influence of travel time changes

	With dynamic improvement							Without dynamic improvement			
σ_{TT}	**OFE**	**OFA**	**TDT**	**TTT**	**ACT**	**AVR**	**ACR**	**OFE**	**OFA**	**TDT**	**TTT**
0	0.4630	0.5230	9915112	274553	406	2.4	12.0	0.4659	0.5309	10131071	280534
300	0.4669	0.5337	10204049	292027	317	2.0	10.8	0.4672	0.5376	10227845	292493
600	0.4650	0.4469	10062157	296084	1665	11.0	41.4	0.4640	0.4502	10056281	296728
900	0.4630	0.4543	9927445	303191	2186	15.6	50.0	0.4700	0.4631	10478748	315267
1200	0.4634	0.4572	9943742	311769	2228	17.0	51.0	0.4706	0.4660	10481892	323281
1500	0.4652	0.4591	10079511	328066	2343	17.0	51.0	0.4720	0.4685	10583907	331568

travel time perturbation described in 7.5.1.2, ranging between 0 and 1500 seconds with an increment of 300 seconds. The results of the simulations are reported in Table 7.11, and a linear regression of the evolution of TTT is represented in Fig. 7.9.

Once again, it is possible to notice that the dynamic routing induces a general improvement in the indicators with respect to the static case, notably an average improvement of 1.83% in total travel time. It is also interesting to note that, when comparing the linear regressions of TTT shown in Fig. 7.9, the slope for the tests with dynamic improvement is less steep than the reference tests. In other words, on top of the improvement in solution quality, the influence of the increase in travel time variation is also mitigated. It can be observed that in this realistic scenario, the computation times for each recalculation problem are always low: they range from about 400 ms when there is no perturbation, to about 2 seconds for the most unstable scenarios (where all clients and vehicles are considered at each step). This demonstrates that the proposed approach can be used in real-time without having the drivers wait for recalculations to occur.

7.6 Conclusion

In this work, it is proposed an innovative strategy to solve the three-dimensional loading capacitated vehicle routing problem with time windows (3L-CVRPTW) which combines two important literature's problems: the Vehicle Routing with Time Windows Problem (VRTWP) and the Container Loading Problem (CLP). The approach may be used for supporting in the

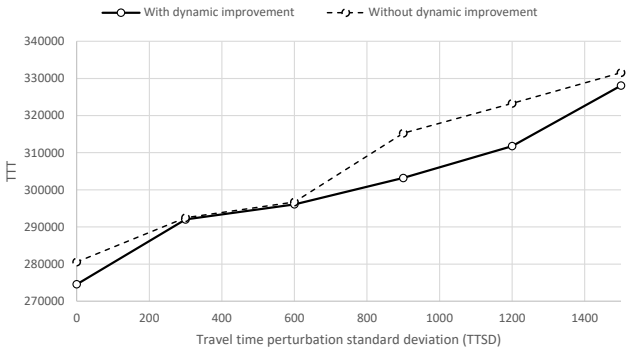


Figure 7.9: Influence of travel time changes - Total travel time (TTT) linear regression

external logistic sector in both the planning operation and the delivery operation in real-time. The proposed approach employs a metaheuristic algorithm composed of two phases: the former is dedicated to the planning of the transport units both in terms of routing and loading of the vehicles and is aimed at solving the 3L-CVRPTW. The latter provides support to the drivers ensuring that the planned routes always constitute the best solution for the shipment, even when during the actual shipment some unexpected disturbances occur. The first phase of the algorithm is implemented by stating three mixed integer linear programming models and heuristics to guarantee that the algorithm addresses the real-field applications. The second one, i.e. the dynamic phase, exploits the potentiality of a genetic algorithm to provide a fast response in real-time. The algorithm is tested both over a literature benchmark and a real dataset provided by an Italian logistics company. The obtained results show that, on the one hand, the proposed approach can provide feasible solutions that minimise travel costs, total travelled distance, and empty space on the vehicles; on the other hand, it can ensure in real-time effective re-routing solutions in case of unexpected events occurring during the delivery. Future works will aim at generalizing the proposed procedure. For instance, additional heuristics could be implemented to further improve the performance of the algorithm. Additionally, further typologies of dynamic events will be considered by the dynamic phase of the proposed algorithm, such as the addition of real-time requests of pick-up and delivery tasks.

References

[1] Zeimpekis, V., Giaglis, G. M., Tatarakis, A., and Minis, I., "Towards a dynamic real-time vehicle management system for urban distribution," *International Journal of Integrated Supply Management*, vol. 3, no. 3, pp. 228–243, 2007.

[2] Pollaris, H., Braekers, K., Caris, A., Janssens, G. K., and Limbourg, S., "Vehicle routing problems with loading constraints: State-of-the-art and future directions," *OR Spectrum*, vol. 37, no. 2, pp. 297–330, 2015.

[3] Moura, A., "A multi-objective genetic algorithm for the vehicle routing with time windows and loading problem," in *Intelligent decision support*, Springer, 2008, pp. 187–201.

[4] Bortfeldt, A. and Homberger, J., "Packing first, routing second—a heuristic for the vehicle routing and loading problem," *Computers & Operations Research*, vol. 40, no. 3, pp. 873–885, 2013.

[5] Smolic-Rocak, N., Bogdan, S., Kovacic, Z., and Petrovic, T., "Time windows based dynamic routing in multi-agv systems," *IEEE Transactions on Automation Science and Engineering*, vol. 7, no. 1, pp. 151–155, 2010. DOI: 10.1109/TASE.2009.2016350.

[6] Pillac, V., Gendreau, M., Guéret, C., and Medaglia, A. L., "A review of dynamic vehicle routing problems," *European Journal of Operational Research*, vol. 225, no. 1, pp. 1–11, 2013.

[7] Psaraftis, H. N., Wen, M., and Kontovas, C. A., "Dynamic vehicle routing problems: Three decades and counting," *Networks*, vol. 67, no. 1, pp. 3–31, 2016.

[8] Facchini, F., Oleśków-Szłapka, J., Ranieri, L., and Urbinati, A., "A maturity model for logistics 4.0: An empirical analysis and a roadmap for future research," *Sustainability*, vol. 12, no. 1, p. 86, 2020.

[9] Tresca, G., Cavone, G., Carli, R., Cerviotti, A., and Dotoli, M., "Automating bin packing: A layer building matheuristics for cost effective logistics," *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 3, pp. 1599–1613, 2022.

[10] Tresca, G., Cavone, G., and Dotoli, M., "Logistics 4.0: A matheuristics for the integrated vehicle routing and container loading problem," in *2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2022, pp. 333–338. DOI: 10.1109/SMC53654.2022.9945179.

[11] Moura, A. and Oliveira, J. F., "An integrated approach to the vehicle routing and container loading problems," *OR spectrum*, vol. 31, no. 4, pp. 775–800, 2009.

[12] Rojas-Cuevas, I.-D., Caballero-Moralesb, S.-O., Sánchez-Partidab, D., and Martínez-Floresb, J.-L., "Three-axes rotation algorithm for the relaxed 3l-cvrp," *Jurnal Kejuruteraan*, vol. 33, no. 1, pp. 63–72, 2021.

[13] Rajaei, M., Moslehi, G., and Reisi-Nafchi, M., "The split heterogeneous vehicle routing problem with three-dimensional loading constraints on a large scale," *European Journal of Operational Research*, vol. 299, no. 2, pp. 706–721, 2022.

[14] Wang, Y., Wei, Y., Wang, X., Wang, Z., and Wang, H., "A clustering-based extended genetic algorithm for the multidepot vehicle routing problem with time windows and three-dimensional loading constraints," *Applied Soft Computing*, vol. 133, p. 109 922, 2023.

[15] Fischetti, M. and Fischetti, M., "Matheuristics," in *Handbook of heuristics*, Springer, 2018, pp. 121–153.

[16] Taş, D., Dellaert, N., Woensel, T. van, and De Kok, T., "The time-dependent vehicle routing problem with soft time windows and stochastic travel times," *Transportation Research Part C: Emerging Technologies*, vol. 48, pp. 66–83, 2014.

[17] Ichoua, S., Gendreau, M., and Potvin, J.-Y., "Vehicle dispatching with time-dependent travel times," *European journal of operational research*, vol. 144, no. 2, pp. 379–396, 2003.

[18] Kok, A. L., Hans, E. W., and Schutten, J. M., "Vehicle routing under time-dependent travel times: The impact of congestion avoidance," *Computers & operations research*, vol. 39, no. 5, pp. 910–918, 2012.

[19] Liu, C., Kou, G., Zhou, X., Peng, Y., Sheng, H., and Alsaadi, F. E., "Time-dependent vehicle routing problem with time windows of city logistics with a congestion avoidance approach," *Knowledge-Based Systems*, vol. 188, p. 104 813, 2020.

[20] Liu, Y., "An optimization-driven dynamic vehicle routing algorithm for on-demand meal delivery using drones," *Computers & Operations Research*, vol. 111, pp. 1–20, 2019.

[21] Okulewicz, M. and Mańdziuk, J., "A metaheuristic approach to solve dynamic vehicle routing problem in continuous search space," *Swarm and Evolutionary Computation*, vol. 48, pp. 44–61, 2019.

[22] Xiang, X., Qiu, J., Xiao, J., and Zhang, X., "Demand coverage diversity based ant colony optimization for dynamic vehicle routing problems," *Engineering Applications of Artificial Intelligence*, vol. 91, p. 103 582, 2020.

[23] Sabar, N. R., Goh, S. L., Turky, A., and Kendall, G., "Population-based iterated local search approach for dynamic vehicle routing problems," *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 4, pp. 2933–2943, 2022. DOI: 10.1109/TASE.2021.3097778.

[24] Johnson, S. C., "Hierarchical clustering schemes," *Psychometrika*, vol. 32, no. 3, pp. 241–254, 1967.

[25] Respen, J., Zufferey, N., and Potvin, J.-Y., "Online vehicle routing and scheduling with continuous vehicle tracking," in *ROADEF-15ème congrès annuel de la Société française de recherche opérationnelle et d'aide à la decision*, 2014.

[26] Pillac, V., Guéret, C., and Medaglia, A. L., "An event-driven optimization framework for dynamic vehicle routing," *Decision Support Systems*, vol. 54, no. 1, pp. 414–423, 2012.

[27] Goldberg, D. E., *Genetic algorithms*. Pearson Education India, 2006.

[28] Goldberg, D. E., Lingle, R., *et al.*, "Alleles, loci, and the traveling salesman problem," in *Proceedings of an international conference on genetic algorithms and their applications*, Carnegie-Mellon University Pittsburgh, PA, vol. 154, 1985, pp. 154–159.

[29] Silva, M. A. L., Souza, S. R. de, Souza, M. J. F., and Bazzan, A. L. C., "A reinforcement learning-based multi-agent framework applied for solving routing and scheduling problems," *Expert Systems with Applications*, vol. 131, pp. 148–171, 2019.

[30] Solomon, M. M., "Algorithms for the vehicle routing and scheduling problems with time window constraints," *Operations research*, vol. 35, no. 2, pp. 254–265, 1987.

[31] *E80 group*, Available on line, 2024. [Online]. Available: https://www.e80group.com/it/.

[32] Mathworks, *Matlab*, Available on line, 2020. [Online]. Available: https://mathworks.com/.

[33] Perron, L. and Furnon, V., *Or-tools*, version v9.7, Google, Aug. 8, 2023. [Online]. Available: https://developers.google.com/optimization/.

[34] Smith, B., *Beginning JSON*. Apress, 2015.

[35] Hasle, G. and Kloster, O., "Industrial vehicle routing," in *Geometric modelling, numerical simulation, and optimization*, Springer, 2007, pp. 397–435.

Chapter 8

Automatic Control of Drones' Missions in a Hybrid Truck-Drone Delivery System

Abstract

Last-mile delivery is one of the most discussed problems of the last decade due to the growing importance of e-commerce and the development of Industry 4.0. In particular, this problem regards the delivery of parcels from the warehouse to the final customers. In order to bring efficiency and innovation, in this chapter a hybrid delivery architecture is considered, which takes advantage of the combined use of a drone and a truck to perform a sequence of pick-ups and deliveries, and the problem of optimal control of the drones' missions is addressed. The reference scenario is the smart city where the drone of the hybrid delivery architecture is in charge of three different pick-up and delivery missions: truck to point (i.e., pick-up from the truck and delivery to the customer), point to point (i.e., delivery to a customer and pick-up from the subsequent customer), and point to truck (i.e., reentry from a customer to the truck). From the control point of view, the drone is optimally guided in all the operating modes, i.e., ascent and descent from/to truck mode, free flight mode with/without payload, and descent for pick-up/delivery mode, by a receding horizon linear quadratic regulator (LQR), which is able to manage the drone in the dynamic landing on a movable vehicle and to allow the changing in real time of the landing point on the truck. Simulation results of the truck-drone delivery architecture are presented and discussed in detail, proving the effectiveness of the proposed method.

Contents

8.1 Introduction

In recent years, logistics (i.e., the set of operations aimed at planning, implementing, and controlling the flow and the storage of goods and related services from external origin points to companies and from companies to consumption points or final customers) is becoming more and more important in the development of the industrial sector [1] and it largely impacts firms' performance [2], [3]. This chapter is focused on Logistics 4.0 (a branch of Industry 4.0) and in particular, on distribution logistics, which generates the highest percentage of the logistic operations costs [4]. One of the most challenging and expensive problems in this field, estimated to range from 13% to 73% of the total distribution costs [5], is the so-called last-mile delivery problem, which consists in the delivery of parcels from the warehouse to the customers (i.e., final

destinations) and whose relevance has grown with the increase of the online commerce and the same-day deliveries to single customers. The main issues that trigger relevance for this problem are [6]: (i) the increasing volume of urbanization and e-commerce, (ii) the sustainability of the shipping process since the rise in urban parcel demands induces a higher number of delivery trucks entering the city centers creating congestion and having negative impacts on health, environment, and safety, (iii) costs, (iv) time pressure because most online retailers sell next- or even same-day deliveries as one of their basic service promises, (v) aging workforce that in many industrialized countries enlarges the problem of employers hiring the required manpower. The last-mile delivery operation is usually performed by humans or vehicles such as vans, bikes, trains, and autonomous vehicles such as autonomous vans and drones.

Innovative applications in the last-mile delivery field come with the use of drones (also known as unmanned aerial vehicles - UAVs) as vehicles. In the last decade, the research and the real field applications of these technologies in the logistic sector are growing exponentially. One of the first applications was developed by Amazon in 2013, with the shipping of small parcels turned into Amazon's drone delivery services Prime Air, i.e., a drone-only delivery system. Subsequently, other companies such as the Workhorse company, the "Project Wing" of Google, and the "Parcelcopter" of DHL proposed a hybrid architecture where drones and trucks cooperate. A typical architecture where a drone and a truck collaborate consists of a drone that autonomously departs from a truck, performs the delivery or the pick-up of the parcel, and then comes back to the truck, while the truck delivers items to the customers or serves as a mobile hub for other drones (in fact, when the drone is on the truck, its battery can be replaced or recharged while waiting for the next trip [7]). The related literature abounds with papers that focus on the design of hybrid drone-truck architectures and their planning. The majority of contributions regard the offline strategic scheduling and routing of trucks and drones in the hybrid truck-drone architecture, while only a few works specifically focus on the online control of the drones' missions in coordination with the truck travel. Therefore, with the aim of bridging this gap, in this chapter, it is proposed a hybrid automated architecture that consists of a truck and a drone, where the missions of the transportation means are coordinated and the drone is optimally guided in real-time by a receding horizon linear quadratic regulator (LQR). The remainder of this chapter is structured as follows. In Section 8.2 the 3D quadrotor with its operating modes and the truck dynamic models are examined. The formulation of the receding horizon LQR controller is presented in Section 8.3, and the simulations setup and results are discussed in Section 8.4. Finally, Section 8.5 reports some concluding remarks.

8.2 System Modelling and Tasks

This section describes the hybrid parcel delivery system based on the combined use of a 3D quadrotor and a truck. In particular, Subsection 8.2.1 and Subsection 8.2.2 present the dynamic models of the quadrotor and the truck, respectively, whereas the quadrotor operating modes during the sequence of pick-ups and deliveries tasks are illustrated in Section 8.2.3.

8.2.1 Quadrotor Dynamics

The quadrotor's space motion can be described through six degrees of freedom (DOF). The space motion consists of three linear movements of the barycenter and three angular movements, namely, three translation and three rotation motions along the three axes that can be controlled by changing the rotational speeds of the four motors. Based on the speed of each propeller, the four basic movements of the quadrotor can be identified: the ascent/descent caused by thrust

due to rotors' rotation, the roll and pitch caused by the difference of the four rotors' thrust, the gravity and the yawing moment caused by the unbalance of the four rotors rotational speeds.

Since the quadrotor is an underactuated non-linear complex system (it has four inputs and six outputs), it is modelled the quadrotor as a rigid body, with a symmetric structure and without ground effect, following the model proposed in [8]. To define the structure and position, two different reference frameworks are considered. It uses the north-east-down (NED) framework for the first inertial coordinate system (fixed), whereas the aircraft body center (ABC) framework is used for the second reference system integral with the quadrotor's barycenter (mobile).

By calling $[x^d, y^d, z^d, \psi^d, \theta^d, \phi^d]^\top$ the vector containing the linear x^d, y^d, z^d and angular ψ^d, θ^d, ϕ^d positions of the quadrotor in the NED framework and $[e^d, v^d, w^d, p^d, q^d, r^d]^\top$ the vector containing the linear e^d, v^d, w^d and angular p^d, q^d, r^d velocities in the ABC framework, it is defined the state vector as follows: $[x^d, y^d, z^d, \psi^d, \theta^d, \phi^d, e^d, v^d, w^d, p^d, q^d, r^d]^\top \in \mathbb{R}^{12}$. Note that the two reference frameworks are linked by the following relation:

$$\boldsymbol{v} = \boldsymbol{R}_m \, \boldsymbol{v}_B \boldsymbol{\omega} = \boldsymbol{T}_m \, \boldsymbol{\omega}_B \tag{8.1}$$

where $\boldsymbol{v} = [\dot{x}^d, \dot{y}^d, \dot{z}^d]^\top \in \mathbb{R}^3$, $\boldsymbol{\omega} = [\dot{\psi}^d, \dot{\theta}^d, \dot{\phi}^d]^\top \in \mathbb{R}^3$, $\boldsymbol{v}_B = [e^d, v^d, w^d]^\top \in \mathbb{R}^3$, $\boldsymbol{\omega}_B = [p^d, q^d, r^d]^\top \in \mathbb{R}^3$, and $\boldsymbol{R}_m$ and $\boldsymbol{T}_m$ are the rotation matrix and the matrix for angular transformations, respectively. Assuming small angles of movement [9], the quadrotor's dynamic model can be simplified by setting $[p^d, q^d, r^d]^\top = [\dot{\psi}^d, \dot{\theta}^d, \dot{\phi}^d]^\top$; hence, it is redefine the state vector in the inertial frame as $\boldsymbol{s}^d = [x^d, y^d, z^d, \psi^d, \theta^d, \phi^d, \dot{x}^d, \dot{y}^d, \dot{z}^d, \dot{\psi}^d, \dot{\theta}^d, \dot{\phi}^d]^\top \in \mathbb{R}^{12}$. As a consequence, by using the state vector $\boldsymbol{s}^d$, the non-linear equations of the quadrotor's dynamics are written in the state space form as:

$$\dot{\boldsymbol{s}}^d = \boldsymbol{f}(\boldsymbol{s}^d) + \sum_{i=1}^{4} \boldsymbol{g}_i(\boldsymbol{s}^d) u^{d,i} \tag{8.2}$$

where $u^{d,1}, u^{d,2}, u^{d,3}, u^{d,4}$ are the four actuators (one for the vertical thrust f_t^d taken upwards and one for each angular motion $\tau_x^d, \tau_y^d, \tau_z^d$) collected in the control input vector $\boldsymbol{u}^d = [f_t^d, \tau_x^d, \tau_y^d, \tau_z^d]^\top \in \mathbb{R}^4$ and:

$$\boldsymbol{f}(\boldsymbol{s}^d) = \begin{bmatrix} \dot{x}^d \\ \dot{y}^d \\ \dot{z}^d \\ \dot{\theta}^d \frac{\text{sen}(\phi^d)}{\cos(\theta^d)} + \dot{\phi}^d \frac{\cos(\phi^d)}{\cos(\theta^d)} \\ \dot{\theta}^d [\cos(\phi^d)] - \dot{\phi}^d [\text{sen}(\phi^d)] \\ \dot{\psi}^d + \dot{\theta}^d [\text{sen}(\phi^d)\tan(\theta^d)] + \dot{\phi}^d [\cos(\phi^d)\tan(\theta^d)] \\ 0 \\ 0 \\ \text{g} \\ \frac{I_{yy} - I_{zz}}{I_{xx}} \dot{\theta}^d \dot{\phi}^d \\ \frac{I_{zz} - I_{xx}}{I_{yy}} \dot{\psi}^d \dot{\phi}^d \\ \frac{I_{xx} - I_{yy}}{I_{zz}} \dot{\psi}^d \dot{\theta}^d \end{bmatrix}$$

$$\boldsymbol{g}_1(\boldsymbol{s}) = [0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; g_1^7 \;\; g_1^8 \;\; g_1^9 \;\; 0 \;\; 0 \;\; 0]^\top$$

$$\boldsymbol{g}_2(\boldsymbol{s}) = [0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; \frac{1}{I_{xx}} \;\; 0 \;\; 0]^\top$$

$$\boldsymbol{g}_3(\boldsymbol{s}) = [0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; \frac{1}{I_{yy}} \;\; 0]^\top$$

$$\boldsymbol{g}_4(\boldsymbol{s}) = [0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; \frac{1}{I_{zz}}]^\top$$

where g is the gravitational acceleration, I_{xx}, I_{yy}, I_{zz} are the components of the diagonal inertia matrix $\boldsymbol{I} = \text{diag}(I_{xx}, I_{yy}, I_{zz})$, and:

$$g_1^7 = -\frac{1}{m^d}[\text{sen}(\phi^d)\text{sen}(\psi^d) + \cos(\phi^d)\cos(\psi^d)\text{sen}(\theta^d)]$$

$$g_1^8 = -\frac{1}{m^d}[\cos(\psi^d)\text{sen}(\phi^d) - \cos(\phi^d)\text{sen}(\psi^d)\text{sen}(\theta^d)]$$

$$g_1^9 = -\frac{1}{m^d}[\cos(\phi^d)\cos(\theta^d)]$$

with m^d is the total mass of the quadrotor. Note that m^d varies according to the presence of the payload: in particular, it holds $m^d = m_0^d$ without payload and $m^d = m_0^d + m^p$ with a payload of mass m^p equal to the weight of the transported item.

Starting from the non-linear quadrotor's dynamics in (8.2), the linearized version is often considered in the literature for control purposes (see Section 8.3). Through the linearization around the nominal point $\boldsymbol{s}^{d,*}$ and control input vector $\boldsymbol{u}^{d,*}$, and by using the sampling time Δt as a time step and the discrete time index n as a subscript of vectors and variables, it is obtained the following discretized linear dynamics:

$$\overline{\boldsymbol{s}}_{n+1}^d = \boldsymbol{A}\overline{\boldsymbol{s}}_n^d + \boldsymbol{B}\overline{\boldsymbol{u}}_n^d \tag{8.3}$$

where:

$$\boldsymbol{A} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & \Delta t & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & \Delta t & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & \Delta t & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \Delta t \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & \Delta t & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & \Delta t & 0 & 0 \\ 0 & 0 & 0 & 0 & -\Delta t\,\text{g} & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -\Delta t\,\text{g} & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\boldsymbol{B} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -\frac{\Delta t}{m^d} & 0 & 0 & 0 \\ 0 & \frac{\Delta t}{I_{xx}} & 0 & 0 \\ 0 & 0 & \frac{\Delta t}{I_{yy}} & 0 \\ 0 & 0 & 0 & \frac{\Delta t}{I_{zz}} \end{bmatrix}_{\boldsymbol{s}^d = \boldsymbol{s}^{d,*}, \boldsymbol{u}^d = \boldsymbol{u}^{d,*}}$$

with $\overline{\boldsymbol{s}}_n^d = \boldsymbol{s}_n^d - \boldsymbol{s}^{d,*}$ and $\overline{\boldsymbol{u}}_n^d = \boldsymbol{u}_n^d - \boldsymbol{u}^{d,*}$.

8.2.2 Truck Model

The truck model is formulated in state space form as:

$$\boldsymbol{s}_{n+1}^t = f(\boldsymbol{s}_n^t, \boldsymbol{u}_n^t) \tag{8.4}$$

where the state vector is defined as $\boldsymbol{s}^t = [x^t, y^t, z^t, \dot{x}^t, \dot{y}^t, \dot{z}^t]^\top \in \mathbb{R}^6$ with x^t, y^t, z^t the truck's positions along the X, Y, Z axes, $\dot{x}^t, \dot{y}^t, \dot{z}^t$ its linear velocities, and the control input vector as $\boldsymbol{u}^t = [\ddot{x}^t, \ddot{y}^t, \ddot{z}^t]^\top \in \mathbb{R}^3$ with $\ddot{x}^t, \ddot{y}^t, \ddot{z}^c$ the truck's accelerations.

8.2.3 Quadrotor Operating Modes

The considered truck-drone delivery system is composed of a truck on which the drone's charging base is positioned and a drone that performs the pick-up and delivery of parcels from/to the customers in the surrounding areas.

For the development of the proposed truck-drone delivery system automation, there are considered three operating modes for the drone that are defined as follows [10].

I) **Ascent from customer and from truck, and descent to truck mode**: Ascent and descent are performed along a vertical and oblique axis, respectively. In particular, for the ascent, the quadrotor starts from the landing point, which is situated near the customer or in the barycentric position of its charging base located on the roof of the truck, and reaches vertically a certain altitude where it begins to hover, while for descent the opposite occurs but in oblique.

II) **Free flight with/without payload mode**: In this operating mode, the quadrotor is in free flight and there is no contact with the road.

III) **Descent for pick-up/delivery mode**: Starting from a certain altitude with a non-zero velocity, the quadrotor descends with a gradually decreasing velocity as it approaches the customer.

8.3 Control Strategy

For all the drone's operating modes described in Section 8.2.3, a receding horizon LQR controller [11] is implemented to control the quadrotor. According to the receding horizon approach, given the sampling time Δt, the optimization problem must be solved iteratively at each $j\Delta t$ time instants, until the end of the delivery mission. It has to be highlighted that the nominal landing point might vary during the mission, since a dynamical platform is considered, thus variables $s_j^{d,*}$ and $u_j^{d,*}$ are updated at each time step j and consequently marked by j as a subscript.

By assuming the lengths of the prediction horizon and control horizon coincident and equal to N, the receding horizon open loop optimization problem at time step j is defined by introducing the following objective function:

$$\begin{aligned} J_{(N)} = &(\boldsymbol{s}_{j+N}^d - \boldsymbol{s}_j^{d,*})^\top \boldsymbol{Q}_{j+N} (\boldsymbol{s}_{j+N}^d - \boldsymbol{s}_j^{d,*}) \\ &+ \sum_{n=0}^{N-1} [(\boldsymbol{s}_{j+n}^d - \boldsymbol{s}_j^{d,*})^\top \boldsymbol{Q}_{j+n} (\boldsymbol{s}_{j+n}^d - \boldsymbol{s}_j^{d,*}) \\ &+ (\boldsymbol{u}_{j+n}^d - \boldsymbol{u}_j^{d,*})^\top \boldsymbol{R}_{j+n} (\boldsymbol{u}_{j+n}^d - \boldsymbol{u}_j^{d,*})]. \end{aligned} \tag{8.5}$$

where $\boldsymbol{Q}_{j+N} \in \mathbb{R}^{12\times12}$, $\boldsymbol{Q}_{j+n} \in \mathbb{R}^{12\times12}$, and $\boldsymbol{R}_{j+n} \in \mathbb{R}^{4\times4}$ are the final cost, state cost, and input cost diagonal matrices to be tuned. Note that the three terms in (8.5) present the final state deviation, state deviation, and input size, respectively.

Given the initial state $\boldsymbol{s}_j^d$, let $\boldsymbol{u}_{j+n}^{LQR}, n = 0, ..., N-1$ be the control sequence that minimizes the quadratic cost function $J_{(N)}$ subject to the state equation:

$$\begin{aligned} &\boldsymbol{s}_{j+n+1}^d = \boldsymbol{A}(\boldsymbol{s}_{j+n}^d - \boldsymbol{s}_j^{d,*}) + \boldsymbol{B}(\boldsymbol{u}_{j+n}^{LQR} - \boldsymbol{u}_j^{d,*}) + \boldsymbol{s}_j^{d,*}, \\ &\forall n = 0, \ldots, N-1. \end{aligned} \tag{8.6}$$

In particular, the optimal control law is computed by the following iterative scheme:

$$\boldsymbol{u}_{j+n}^{LQR} = \boldsymbol{K}_{j+n}(\boldsymbol{s}_{j+n}^d - \boldsymbol{s}_j^{d,*}) + \boldsymbol{u}_j^{d,*}, \forall n = 0, \ldots, N-1 \tag{8.7}$$

where the state $\boldsymbol{s}_{j+n}^d$ is updated in accordance with the model in (8.6) and the feedback gain $\boldsymbol{K}_{j+n} \in \mathbb{R}^{12\times 12}$ is obtained through the following well-known Riccati difference equations [12] that are solved recursively backwards:

$$\begin{aligned} &\boldsymbol{K}_{j+n} = -(\boldsymbol{R}_{j+n} + \boldsymbol{B}_j^\top \boldsymbol{P}_{j+n+1}\boldsymbol{B}_j)^{-1}\boldsymbol{B}_j^\top \boldsymbol{P}_{j+n+1}\boldsymbol{A}_j, \\ &\boldsymbol{P}_{j+n} = \boldsymbol{Q}_{j+n} + \boldsymbol{A}_j^\top \boldsymbol{P}_{j+n+1}\boldsymbol{A}_j + \boldsymbol{A}_j^\top \boldsymbol{P}_{j+n+1}\boldsymbol{B}_j\boldsymbol{K}_{j+n} \\ &\forall n = 0, \ldots, N-1 \end{aligned} \tag{8.8}$$

being $\boldsymbol{P}_{j+n} \in \mathbb{R}^{12\times 12}$ the parameter matrix and initializing $\boldsymbol{P}_{j+N} = \boldsymbol{Q}_{j+N}$.

The receding horizon policy proceeds by implementing only the first control input vector $\boldsymbol{u}_j^{LQR}$, whilst the rest of the control sequence $\boldsymbol{u}_{j+n}^{LQR} \quad \forall n = 1, \ldots, N-1$ is not considered and $\boldsymbol{s}_{j+1}^d$ is employed to update the optimization problem as a new initial condition. The algorithm proceeds until the end of the delivery mission, by shifting the horizon ahead by one time step

8.4 Experimental Results

In this section, it is described the system setup and the simulation results of the proposed real-time control strategy for a hybrid truck-drone delivery system. It is highlighted that the quadrotor's control system is implemented on a Jupyter Notebook.

8.4.1 System Setup

In the context of the last-mile delivery problem, i.e., delivery of items from the warehouse to the customers, the goal of our experiment is to efficiently perform a sequence of pick-up and delivery of parcels tasks in a smart city, by employing a hybrid truck-drone delivery architecture composed of a truck and a drone. Offline scheduled missions and depart/return together from/to the warehouse – where the truck is loaded with both its parcels and the ones of the drone – are assigned to the truck and the drone. The drone can recharge on the truck roof and must pick up and release from/to the truck light parcels, depending on its admissible payload. Differently, the truck is devoted to the delivery of heavier parcels. Pick-ups and deliveries can be assigned to both the truck and the drone, but at each mission, the truck departs and returns from/to the warehouse, while the drone departs/returns from/to the moving truck. Thanks to state-of-the-art sensors mounted on drones board that allow collecting data quickly and easily, the control station can communicate with the drone by notifying the trajectory (both position and velocity) of the truck in accordance with a certain sampling time Δt and thus, drone and truck rejoin along the fixed route of the truck.

More specifically, with the use of the three quadrotor's operating modes listed in Section 8.2.3, i.e., ascent and descent from/to truck mode, free flight mode with/without payload, and descent

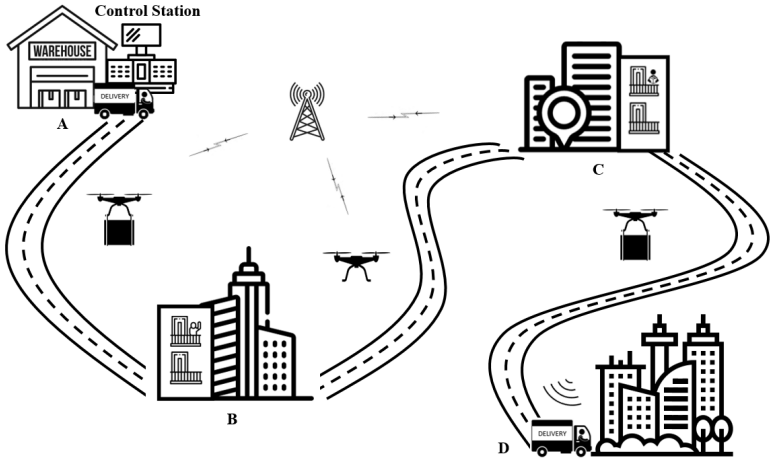


Figure 8.1: Parcels' last-mile delivery architecture with operating tasks phases.

Table 8.1: Quadrotor dynamics parameters.

Phase	m^d **[kg]**	$\boldsymbol{I}$ **[kg m^2]**
Truck (A) to Point (B) Point (C) to Truck (D)	2.18	diag(0.0087,0.0087,0.0123)
Point (B) to Point (C)	1.38	diag(0.0037,0.0037,0.0073)

for pick-up/delivery mode, it is possible to perform a mission (see Fig. 8.1) through three phases, as detailed below.

1) **Truck (A) to Point (B)**: In this phase, once the truck has left the warehouse, the drone ascends from its charging base located on the roof of the truck with the parcel directed towards the customer placed at point B. Hence, the drone performs the route in free flight mode with payload and then executes the descent towards the customer.

2) **Point (B) to Point (C)**: In this phase, after the drone has released the parcel to the customer located at point (B), it leaves in free flight mode without payload towards the second customer, i.e., point (C).

3) **Point (C) to Truck (D)**: In this phase, after the drone picks up the customer parcel located at point (C), it receives the trajectory (both position and velocity) of the truck from the control station. Thus, the drone follows the trajectory of the truck in free flight mode with payload, and then, once reached, it is ready to descend towards the landing point situated in the barycentric position of the charging base, i.e., point (D). Note that, since the truck is moving, the drone is initially aligned with the truck along the X and Y axes, while keeping a given offset along Z, and reaches the landing point from behind. As a final remark, during the current phase, the truck, if necessary, slows down its velocity to adapt to the technical characteristics of the drone. The same holds for the drone as well. The realistic scenario addressed in this chapter is shown in Fig. 8.1, which reproduces the hybrid movable architecture that consists of a drone and a truck. In particular, Fig. 8.1 illustrates a portion of an entire daily truck-drone mission, i.e., the route followed by the truck transporting the items from the warehouse to the various customers located in different places and the drone that helps the courier to perform pick-ups and deliveries and once the assigned tasks are completed, it intercepts the truck on which the charging base

is placed. The experiment is conducted considering the well-known DJI Phantom 4 Pro [13] drone, which has a maximum speed of 72 km/h and a flight autonomy of about 30 minutes. In particular, the quadrotor is modeled in accordance with the dynamic parameters in Table 8.1, where m^d and $\boldsymbol{I}$ indicate the total mass of the quadrotor and the diagonal inertia matrix, respectively. The load of mass m^p carried by the vacuum gripper attached to the quadrotor base is equal to 0.8 kg. Instead, the drone' control input vector is defined as $\boldsymbol{u}^d = [m^d g, 0, 0, 0]$ with the gravitational acceleration g set to 9.81 m/s^2. To conclude the system setup, it is set the sampling time $\Delta t = 0.01$ s, $N = 2000$, as representing a good compromise between computational complexity and solution quality, the initial state cost matrix $\boldsymbol{Q}_{j+N} = 200\, \boldsymbol{I}_{12}$, and the initial input cost matrix $\boldsymbol{R}_{j+N} = 2\, \boldsymbol{I}_4$ computed for each time step j.

8.4.2 Results

The goal of this work is to control a drone employed in the last-mile delivery problem in tandem with a truck with the aim of performing pick-ups/deliveries from/to customers in a smart city. In the proposed model, the truck works as a primary vehicle and follows a fixed route determined offline before the beginning of the mission. Instead, the drone departs from the roof of the truck (which is following its fixed route) and visits the customers according to the schedule. Then, in order to take other parcels from the truck and be ready for the next sortie, it returns to the moving truck whose position is notified by the control station.

The parcels' last-mile delivery architecture with the truck-drone combined operations is represented in a schematic configuration in Fig. 8.1 whereas in a 3D reconstruction in Fig. 8.2 to help the reader imagine the real scenario of an urban environment. For the sake of clarity, it highlighted that the nodes (A), (B), (C), and (D) in Fig. 8.1 and Fig. 8.2 are coincident. As can be seen from the 3D view (Fig. 8.2), the truck leaves the warehouse to perform the scheduled deliveries of the heavier parcels and at the same time the drone carries out pick-ups and deliveries of parcels (i.e., Point (B) and Point (C)) with a lower payload in the surrounding areas. Furthermore, Fig. 8.2 shows the trajectory followed by the drone to chase and catch up with the truck and then to perform the descent towards the landing point located on the roof of the moving truck. The trajectory from Point (C) to Truck (D) is also represented in red in Fig. 8.3, where it is possible to observe the perfect tracking executed by the drone of the truck's trajectory (in blue) given at each sampling time Δt by the control station in terms of position and velocity.

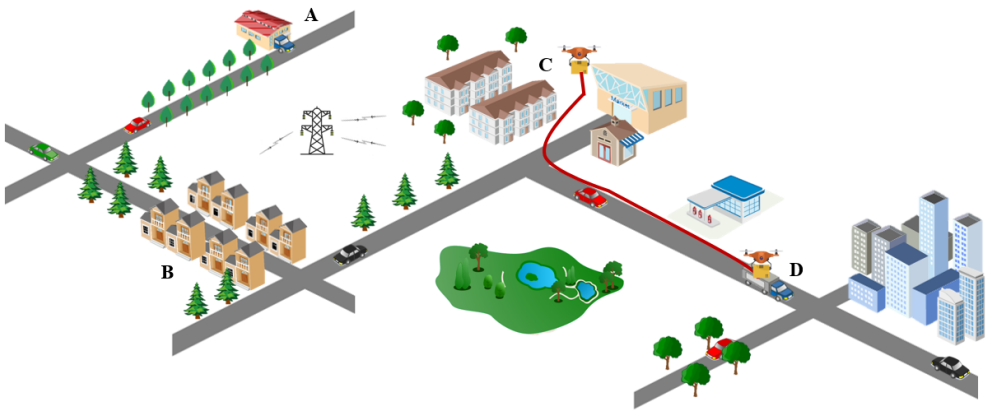


Figure 8.2: Last-mile delivery of parcels architecture with focus on the trajectory performed by the drone in the third operating task phase, i.e., Point (C) to Truck (D).

The effectiveness of the implemented controller, i.e., the receding horizon LQR, lies in the possibility of changing online the position of the landing point located on the roof of the truck (i.e., the final cost in the objective function (8.5)). The drone can not only vary its speed, depending on the technical characteristics of the truck and vice-versa, but it can also change its trajectory towards the landing point in case there is an unexpected event, such as a slowdown of the truck due to traffic or merely a transmission error by the control station. Fig. 8.4 illustrates the trajectory followed by the drone as the position of the landing point varies from Point (D) to Point (D') and then to Point (D"). More specifically, it is possible to see the drone's predicted routes in red from different starting points placed forward on the given prediction horizon whereas the drone's actual route in green from Point (C) to Truck (D"), which intersects the n-th predicted routes.

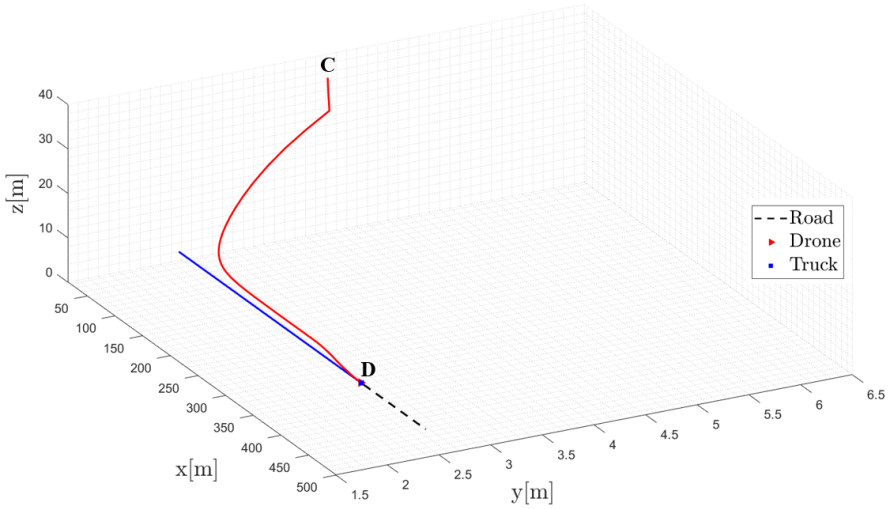


Figure 8.3: Drone's trajectory from Point (C) to Truck (D).

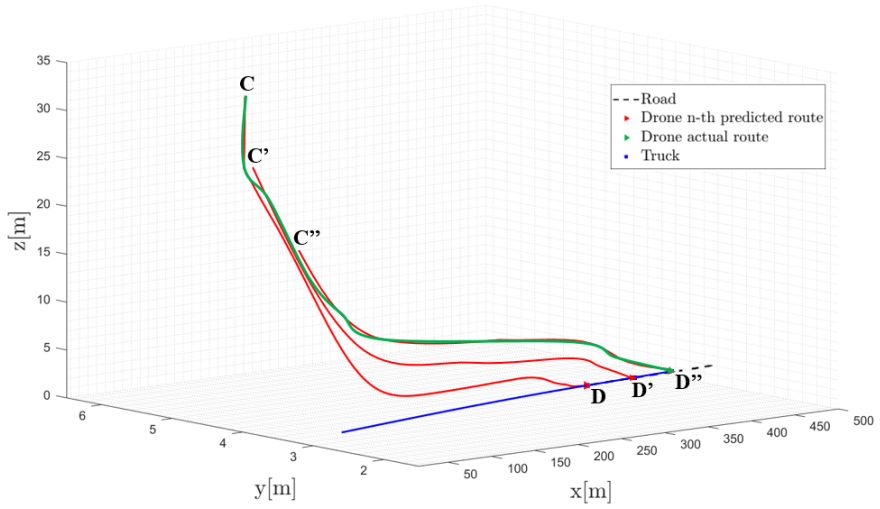


Figure 8.4: Drone's trajectory variation with moving landing point located on the truck.

8.5 Conclusion

This chapter presents an automatic real-time control approach for a hybrid truck-drone delivery system devoted to last-mile deliveries. In particular, the drone is used to help the courier to perform a sequence of pick-ups and deliveries of parcels from/to the customers in the surrounding areas of the smart city and, once the scheduled tasks are finalized, it intercepts the moving truck and descends towards the charging base placed on its roof. To accomplish the desired mission, the drone is optimally guided by a receding horizon linear quadratic regulator in all its operating modes, which are classified as: ascent and descent from/to truck mode, free flight with/without payload mode, and descent for pick-up/delivery mode. In particular, the controller is able to manage in real-time the drone's landing on the moving truck and allow the online change of the landing point on the truck.

Future works will focus on enhancing the dynamical model of the drone, in order to consider the effects of the terrain and the airflow generated by the propellers, and on employing a dynamical model of the truck with the aim of enhancing the estimation of the landing point's position where the descent takes place. In addition, it will be useful to include the energy management objective in the current cost function, to compare the performance of the optimal control technique considered in this chapter with other receding horizon control strategies like model predictive control with constraints on the translational speed of the drone and on the flying elevation in a city environment or some visual-based control approaches and to implement the proposed architecture on a real system.

References

[1] Winkelhaus, S. and Grosse, E. H., "Logistics 4.0: A systematic review towards a new logistics system," *International Journal of Production Research*, vol. 58, no. 1, pp. 18–43, 2020.

[2] Bag, S., Gupta, S., and Luo, Z., "Examining the role of logistics 4.0 enabled dynamic capabilities on firm performance," *The International Journal of Logistics Management*, 2020.

[3] Boenzi, F., Digiesi, S., Facchini, F., Mossa, G., and Mummolo, G., "Sustainable warehouse logistics: A nip model for non-road vehicles and storage configuration selection," *Proceedings of the XX Summer School Operational Excellence Experience "Francesco Turco*, 2015.

[4] Tresca, G., Cavone, G., Carli, R., Cerviotti, A., and Dotoli, M., "Automating bin packing: A layer building matheuristics for cost effective logistics," *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 3, pp. 1599–1613, 2022.

[5] Cavone, G., Epicoco, N., Carli, R., Del Zotti, A., Paulo Ribeiro Pereira, J., and Dotoli, M., "Parcel delivery with drones: Multi-criteria analysis of trendy system architectures," in *2021 29th Mediterranean Conference on Control and Automation (MED)*, 2021, pp. 693–698. DOI: 10.1109/MED51440.2021.9480332.

[6] Boysen, N., Fedtke, S., and Schwerdfeger, S., "Last-mile delivery concepts: A survey from an operational research perspective," *Or Spectrum*, vol. 43, no. 1, pp. 1–58, 2021.

[7] Chung, S. H., Sah, B., and Lee, J., "Optimization for drone and drone-truck combined operations: A review of the state of the art and future directions," *Computers & Operations Research*, vol. 123, p. 105 004, 2020. DOI: https://doi.org/10.1016/j.cor.2020.105004. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0305054820301210.

[8] Lee, D., Burg, T. C., Dawson, D. M., Shu, D., Xian, B., and Tatlicioglu, E., "Robust tracking control of an underactuated quadrotor aerial-robot based on a parametric uncertain model," in *2009 IEEE international conference on systems, man and cybernetics*, IEEE, 2009, pp. 3187–3192.

[9] Das, A., Subbarao, K., and Lewis, F., "Dynamic inversion with zero-dynamics stabilisation for quadrotor control," *IET control theory & applications*, vol. 3, no. 3, pp. 303–314, 2009.

[10] Proia, S., Cavone, G., Camposeo, A., Ceglie, F., Carli, R., and Dotoli, M., "Safe and ergonomic human-drone interaction in warehouses," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2022, pp. 6681–6686.

[11] Primbs, J. A. and Nevistic, V., "Constrained finite receding horizon linear quadratic control," in *Proceedings of the 36th IEEE Conference on Decision and Control*, IEEE, vol. 4, 1997, pp. 3196–3201.

[12] Bittanti, S., Laub, A. J., and Willems, J. C., *The Riccati Equation*. Springer Science & Business Media, 2012.

[13] *Phantom 4 Pro*, https://www.dji.com/it/phantom-4-pro/info, Accessed: 2022-09-01.

Chapter 9

Conclusions

In conclusion, this doctoral thesis thoroughly investigates the profound impact of Industry 4.0 and Logistics 4.0 for both companies and academia, providing substantial theoretical and practical advancements for both actors. As a matter of fact, the advent of Industry 4.0, with its technologies such as the Internet of Things (IoT), Big Data, Artificial Intelligence, and automation, has brought a profound shift in the way industrial processes are conceived and executed. Simultaneously, also thanks to the advent of the e-commerce, Logistics 4.0 has risen to prominence as a vital component of this transformation, with a focus on optimizing the entire logistics and supply chain spectrum.

This work explores multiple aspects of such revolution, representing a comprehensive journey across multiple dimensions of the transformative Industry 4.0 and Logistics 4.0 landscape in the control and automation perspective. A significant contribution of this research regards the application of matheuristic approaches to address critical challenges within the realm of logistics. In particular, in the intricate field of logistic complexities, here the focus is on the modeling and solution of the 3D Bin Packing Problem, optimizing the arrangement of items within confined spaces and fulfilling logistic physical constraints that take into account also the robotization and automation of the packing process. The investigation extends to the design of Vertical Lift Modules (VLM) warehouses, where innovative strategies are developed to maximize efficiency in the allocation of items into trays. Furthermore, the study dives into the intricacies of the multi-drop multi-container loading problem, unraveling the complexities associated with packing multiple bins for diverse deliveries. Additionally, the integrated vehicle routing and container loading problem are analysed, with a focus on formulating solutions that synchronize the routing of vehicles with the loading of containers, creating a harmonized approach to logistics optimization. The proposed solutions provide not only theoretical insights but also practical tools for enhancing the efficiency and effectiveness of logistics operations. Moreover, the study has touched upon the real-time control of hybrid truck-drone delivery systems, reflecting the trend toward autonomous and sustainable last-mile logistics. The potential of such systems, as demonstrated by this research, highlights the ever-increasing convergence of technology and logistics, enabling more agile, cost-effective, and environmentally friendly delivery methods.

In summary, it is evident that Industry 4.0 and Logistics 4.0 are symbiotic forces shaping the future of manufacturing and supply chain management. They offer the promise of enhanced productivity, sustainability, and customer satisfaction. Nevertheless, this transformation comes with its own set of challenges, from workforce adaptation to resource management improvement, which need to be diligently addressed. These challenges demand meticulous attention and strategic solutions to ensure a smooth transition towards the future paradigm.

In the years to come, future research in this field should continue to bridge the gap between theory and practice, ensuring that the theoretical models and matheuristic approaches developed are not just academic advancements but practical solutions that can be readily adopted by logistics and manufacturing industries to advance their automation. Furthermore, the dynamic evolution of these technologies signals the need for continuous adaptation and the development of novel methods and strategies. This adaptive mindset becomes especially critical in anticipation of the imminent Industry 5.0, where the ever-evolving landscape will demand innovative solutions to address emerging challenges and opportunities. The studies conducted in this thesis unveil

numerous avenues for potential improvements and future developments. One notable direction involves the incorporation of additional logistic constraints, such as the intriguing possibility of considering multiple bin sizes in the context of the bin packing problem, or also the opportunity of enhancing the stability of solutions within the container loading problem, refining the optimization of container configurations for increased robustness. Furthermore, there is ample room for advancement in the tackled problem, such as the integration of a "sequencing" model for the definition of the placement order of the items inside a bin, or also the exploration of new dimensions in the dynamic vehicle routing, incorporating various typologies of dynamic events. A common improvement to all the topics discussed is the application of the latest machine learning and artificial intelligence techniques. The integration of these cutting-edge technologies holds the potential to yield solutions that are not only more accurate but also faster, aligning seamlessly with the dynamic nature of real-time logistics operations. As the logistics and supply chain landscape evolves, the strategic incorporation of these advanced techniques stands as a pivotal step toward achieving systems that are not only more efficient but also adaptive and intelligent.

In the context of Industry 5.0, this thesis is just the starting point of a revolution where the landscape is poised for a new wave of transformation characterized by even greater connectivity, collaboration, and intelligence. As a matter of fact, Industry 5.0 envisions a highly integrated and cooperative environment where humans and machines work in tandem, leveraging advanced technologies such as the Internet of Things, Artificial Intelligence, and Cyber-Physical Systems. This era holds the promise of further optimizing production processes, enabling more flexible and personalized manufacturing, and fostering a heightened level of adaptability to dynamic market demands. As the industry advances into this new frontier, research efforts must proactively address the challenges associated with increased connectivity, cybersecurity, and the evolving role of the workforce in a highly automated and intelligent manufacturing ecosystem. By embracing an adaptive mindset and fostering innovation, the transition to Industry 5.0 can be navigated effectively, unlocking new potentials for the manufacturing and logistics landscape.