Palletization and Build-up Scheduling Problem from an Air Cargo Application

Mengning Mao

Delft University of Technology

TU Delft

Palletization and Build-up Scheduling Problem from an Air Cargo Application

by

Mengning Mao

Student number: 5683130
Thesis duration: April 2, 2024 – December 12, 2024
Thesis supervisor: Dr. S. (Stefano) Fazi, TU Delft
Dr. A. (Alessandro) Bombelli, TU Delft

Cover: AAPA: e-commerce shipping drives year-end air cargo demand by Payload Asia, 2024
Style: TU Delft Report Style, with modifications by Daan Zwaneveld

TU Delft

Preface

This research stems from a profound interest in addressing the evolving challenges of global logistics and supply chain management, particularly in the context of air cargo operations. The increasing complexity of cross-border e-commerce logistics and the rising demand for efficient, timely, and cost-effective solutions highlight the critical need for optimization in both strategic planning and operational execution. This thesis investigates how advanced mathematical modeling and heuristic algorithms can be applied to streamline logistical processes, thereby contributing to a more integrated and efficient global supply chain.

The study began with an exploration of the intricate relationship between packing efficiency and scheduling in air cargo operations. It identified key gaps in existing methodologies, particularly the lack of frameworks that holistically address the interplay between spatial optimization and temporal constraints. Building on these observations, this research proposes a two-stage framework that integrates the Air Cargo Palletization Problem and the Build-up Scheduling Problem. The framework leverages mathematical and heuristic models to optimize cargo packing and scheduling, ensuring the seamless alignment of these two critical logistical processes.

This work would not have been possible without the guidance and support of many individuals. I am profoundly grateful to my thesis supervisors, Alessandro Bombelli and Stefano Fazi, for their invaluable feedback, mentorship, and intellectual guidance throughout this journey. Their expertise and encouragement have been instrumental in shaping this research and bringing it to fruition. In addition, I would like to show gratitude to Cainiao for providing the operational data needed for this work.

This thesis is dedicated to my family and friends, whose steadfast support and encouragement have been my foundation throughout this academic endeavor. Despite the physical distance from the Netherlands, my family's unwavering belief in me, along with their emotional and financial sacrifices, has been an enduring source of strength and motivation. I am especially grateful to my parents, who stood by me through every challenge, offering their unconditional love and support. Their sacrifices and faith in my abilities have made this journey possible.

As the logistics industry continues to evolve, I hope this research contributes meaningfully to the ongoing discourse on optimizing air cargo operations. By addressing the challenges and opportunities in palletization and scheduling, I aspire to foster the development of more efficient and sustainable logistics systems. May this work pave the way for a future where technological innovation seamlessly integrates with operational excellence and upholds the human values that underpin progress.

Mengning Mao
Rotterdam, December 2024

Abstract

With the rapid development of cross-border e-commerce logistics, how to efficiently load goods into Unit Loaders (ULDs) and ensure their on-time delivery has become a key issue in logistics systems. Based on Cainiao's actual logistics challenges, this paper proposes a comprehensive loading planning scheme in two phases: the first phase solves how each item can be efficiently packed into ULDs in 3D space, and the second phase solves when each ULD can be loaded on multiple parallel workstations. This design-oriented approach fine-tunes and integrates existing optimization techniques into a cohesive pipeline to tackle these interconnected problems systematically.

To tackle the 3D Bin Packing Problem (3DBPP), two approaches, Mixed Integer Programming (MIP) and Extreme Point Heuristic (EPH), are used in this paper. The MIP model maximizes space utilization through accurate optimization and is suitable for small-scale packing scenarios, while the EPH algorithm performs well in large-scale scenarios and generates high-quality approximate solutions in a short period. Although its space utilization is slightly lower than that of MIP, its solution efficiency is well suited to real logistics operations that require fast response time.

For the Build-up Scheduling Problem (BSP), a parallel machine scheduling model is used to optimize the assembly sequence and timing of ULDs to ensure that all ULDs can be loaded within a strict time window. Experimental results show that the model performs well in optimizing workstation load balancing and avoiding delays, which can significantly improve the scheduling efficiency of the whole system.

The research results in this paper are validated to show that the proposed two-stage framework has significant application value in improving space utilization, reducing cargo delays, and optimizing workstation scheduling in real logistics scenarios of Cainiao. Future research can introduce a feedback loop between the two phases, combine real-time data with dynamic adjustment strategies, hybrid algorithms, and other methods to further improve the adaptability and efficiency of the model.

Contents

1 Introduction

1.1. Background

Over the past few decades, air cargo transportation has assumed a pivotal role in global trade. According to the International Air Transport Association (IATA), while air cargo volumes account for only about 1% of total global trade, they amount to $6 trillion in value terms or 35% of total global trade [24]. In particular, air cargo revenues reach 17% of total airline revenues in 2022, much higher than 12% in 2019 [23]. This data highlights the growing importance of air cargo as a source of revenue for the airline industry. Meanwhile, Cainiao, one of the world's largest cross-border e-commerce logistics companies, has a particularly strong presence in international logistics, with more than 1.5 billion cross-border parcels in Fiscal Year 2023, serving more than 100,000 merchants and brands, and providing full-link Business to Business(B2B)/Business to Consumer(B2C) logistics services to help global brands quickly enter the Chinese retail market [10].

A major challenge in the air cargo sector is how to efficiently load cargo of different shapes and sizes into different unit loading devices (ULDs) for subsequent loading into the cargo hold of an aircraft. These ULDs are component assemblies made of containers or net-covered pallets designed to provide standardized-size units for single pieces of baggage or cargo while allowing for rapid loading and unloading [30]. The ULDs have specific shapes and sizes to fit into the lower deck or belly space of an airliner, and in the case of an all-cargo aircraft, the main deck and lower deck [17]. Due to the limited cargo capacity of passenger aircraft and the variability in both supply and demand, it is particularly important to have a sound cargo loading plan. A critical issue in this process is how to best manage cargo allocation to the ULD and placement within it to minimize transport costs and delivery delay penalties. This issue, also known as the Air Cargo Palletization Problem (APP), is one of the most important optimization challenges [48]. Next, building ULDs needs to be carried out at the workstation. Since workstation resources are limited and cargo release and due dates vary, choosing when to assemble and load which cargo onto the ULDs is critical to the smooth operation of the air cargo terminal. Thus the second task is to determine the time to build each ULD and assemble the workstations, i.e., solving the Build-up Scheduling Problem (BSP) [9].

1.2. Objectives

The motivation for this research stems from the practical challenges that Cainiao has encountered in the field of airfreight logistics, namely how to find a balance between transport efficiency (minimizing ULD wastage) and customer satisfaction (meeting delivery deadlines for

orders). This involved not only how to allocate goods of various shapes and sizes to the limited ULD space to maximize space utilization, named Air Cargo Palletization Problem (APP), but also how to sequence and time the loading of goods to ensure on-time delivery and avoid the costs of lateness. Another important aspect is determining when and at which assembly workstation these ULDs are created, which is called the Build-up Scheduling Problem (BSP). Together, these issues define the core optimization challenges of Cainiao Air Cargo Logistics. As a result, Cainiao needed a load plan that could respond quickly to different cargo needs and could process orders efficiently while keeping costs reasonable. We first propose a model incorporating time constraints for the APP encountered by Cainiao in real-world logistics applications, aiming at solving the problem of processing a specified batch of goods before the deadline, introducing a new dimension of time to the traditional 3D Bin Packing Problem. Secondly, we consider the limited resources of workstations, and the next model is proposed to determine the loading order and time of ULD on parallel workstations. The development of these two sequential models will improve the efficiency of loading goods in ULDs while ensuring that customers' time requirements and cost control objectives can be met.

1.3. Research questions

The above objectives and context are achieved by answering the following main research question:

How can the process of loading cargo on ULDs be optimized to improve space utilization and transport efficiency, while ensuring that the air cargo is shipped on time to meet customer demand?

The main research question is jointly answered by the following sub-questions:

- *What models or algorithms exist to pack items in the ULD?*
- *What methods or algorithms exist to consider the time factor in bin packing problems?*
- *How can a model be designed to pack different shapes and sizes of goods in the ULD?*
- *How can the model be designed to constrain the ULD to be built within the item's required timeframe?*
- *How to find a balance between improving space utilization in ULDs and ensuring on-time shipments?*

1.4. Outline of the structure

This paper presents a proposal for the above objective. Firstly, the chapter 2 presents the definition of the problem. In the chapter 3, the existing literature concerning the topic and identifies multiple knowledge gaps are explored. The section 3.3 wraps up the review with a conclusion on the current knowledge and gaps therein. The chapter 4 details the models developed for the Air Cargo Palletization Problem and Build-up Scheduling Problem. Then, the results and performance of the proposed models are tested through real data cases in the chapter 5. Finally, chapter 6 provides conclusions and recommendations for future work.

2

Problem definition

This paper focuses on the build-up loading process within load planning, covering decisions on two problems [9]:

1. How to pack items on ULD?
2. When should ULDs be built?

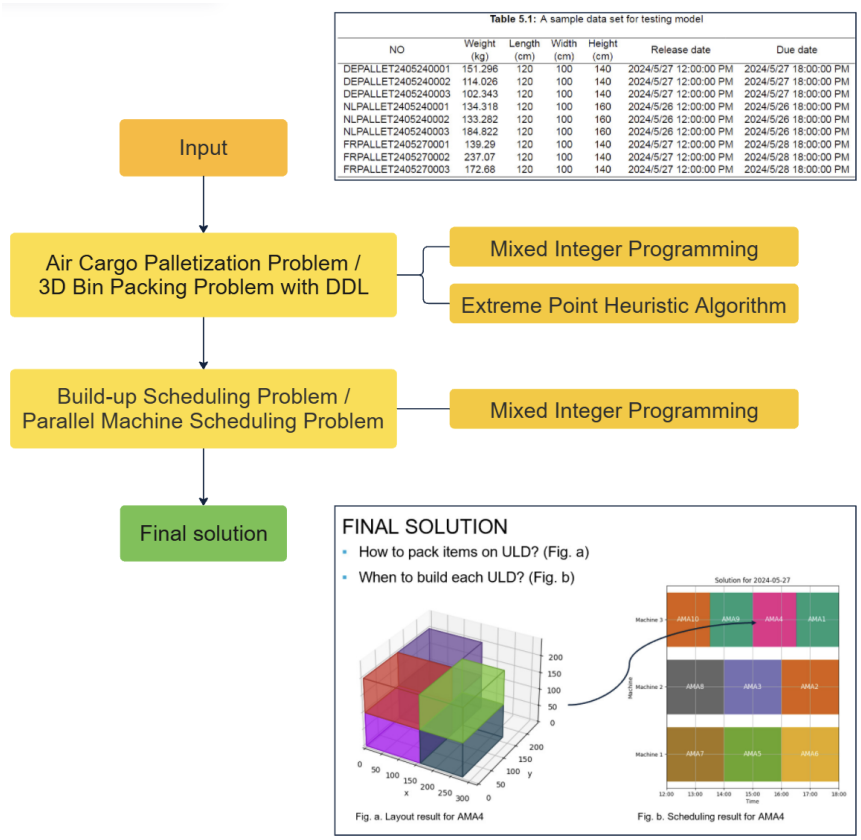


Figure 2.1: Two-Stage Solution Framework for Air Cargo Build-Up Loading Process

The framework depicted in Figure 2.1 outlines the two-stage approach for solving these problems, designed to address the complex challenges of air cargo palletization and build-up scheduling. This work primarily focuses on the development and integration of an effective

framework rather than the creation of novel algorithms or theoretical breakthroughs. By combining and adapting existing models and methodologies, this framework provides a systematic solution to the real-world problems faced in air cargo operations.

The first decision corresponds to the Air Cargo Palletization Problem, which is modeled as a 3D Bin Packing Problem with Due Date (3DBPP-DDL). This stage focuses on determining the optimal way to pack items into ULDs, taking into account both geometric constraints and time-dependent factors, such as delivery deadlines. To solve this, a Mixed Integer Programming (MIP) model is used, alongside an Extreme Point Heuristic (EPH) algorithm, to ensure that the available ULD space is efficiently utilized while minimizing any potential delays.

The second decision corresponds to the Build-up Scheduling Problem, which not only finds the optimal ULD construction sequence for each workstation but also considers finding the optimal workstation allocation in the case of multiple workstations. The scheduling objective is to minimize makespan and ensure that ULDs are packed and handed over on time, adhering to both processing times and delivery deadlines [38].

These two stages work sequentially to solve the overarching problem of the build-up loading process. The result is a comprehensive loading plan, which provides: an efficient packing strategy for items on ULDs (Fig. a), and a detailed schedule indicating when each ULD should be processed and built (Fig. b).

2.1. Air Cargo Palletization Problem

From the perspective of the Variable-Sized Bin Packing Problem (VS-BPP), optimizing the allocation of goods to ULDs can reduce the number of ULDs needed, thereby cutting costs. However, loading must also account for time constraints due to scheduled flight departures, with the goal of meeting delivery deadlines. Failure to meet these deadlines may result in shipment or ULD delays to subsequent flights, potentially resulting in penalties for delayed deliveries to customers.

This paper addresses Air Cargo Palletization Problem as a 3D Bin Packing Problem with deadlines. Given ULD dimensions and various sizes and weights of cargo, the goal is to find a loading layout that maximizes container space utilization, considering practical loading constraints. The paper takes into account the limited availability of different ULD types at airports, as airlines may not supply an unlimited number of all ULD types per aircraft at all times. Additionally, each item's issue time and deadline are considered. Only available cargo after an order has been placed can be loaded onto ULDs, with each cargo having the same assembly time and a deadline (DDL) dictating the ideal completion and handover to the airline. Additionally, the model accounts for potential delays in the loading process, which could result in missed deadlines. The problem is shown below:

min	unused space of ULDs and delay penalties of cargos
s.t.	ULD allocation
	ULD weight limitations
	item positioning within ULD
	item availability
	item due date

2.2. Build-up Scheduling Problem

The build-up scheduling problem is defined as a Parallel Machine Scheduling Problem (PMSP). There are numerous types of PMSP, and our work concentrates on the construction of ULDs in

an airline's cargo terminal. In this research, we assume that given a finite number of identical workstations on which ULDs can be placed, they are always available from time zero and can only build one ULD at a time. These workstations are set up to process all of the required jobs, which are the ULD outputs from the prior problem. A job becomes available for processing when it is released, and each job has a processing time. This paper assumes that ULDs should be built entirely on a single workstation and that building cannot be halted once begun, i.e., preemption is not permitted. Furthermore, the construction timeframe must begin on or after the maximum release date specified in the ULD and end before the item's minimum deadline. Within this time frame, a machine allocation can be identified that ensures no workbench builds more than one ULD at a time and no ULD begins processing before its contents are released. The final completion time is the important KPI that outlines when the entire build-up process ends [1]. The problem is shown below:

min	overall completion time
s.t.	workstation allocation
	completion time
	order of precedence
	no overlap
	time windows

3

Literature review

3.1. Air Cargo Palletization Problem

The Air Cargo Palletization Problem is actually an application of the Bin Packing Problem (BPP) that considers the characteristics of air cargo. The Bin Packing problem has been studied since the early 1970s [19] and is a fundamental problem in operations research, computational geometry, and supply chain management, often revolving around optimizing the placement of objects of various sizes into a finite number of containers or ”bins” to minimize the number of bins used [48].

3.1.1. One-dimensional Bin Packing Problem (1DBPP)

The bin packing problem can be traced back to the one-dimensional bin packing problem, where a set of objects of different sizes must be packed into a minimum number of identical bins [36]. 1DBPP formulas are commonly utilized to model time-sensitive situations (such as scheduling). Packing difficulties with due dates, for example, are increasingly being researched and have crucial practical implications [20]. Table 3.1 summarises previous literature on one-dimensional packing problems that take time factors into account.

Table 3.1: Recent literature on 1DBPP addressing time constraints

Publications	Type	Availability	Due date	Objective	Allow delays?
Reinertsen and Vossen (2010)	1DCSP	-	x	min stock lengths + tardiness	y
Fazi, Van Woensel, and Fransoo (2012)	VSBPP	x	x	min costs + waiting times + tardiness	y
Polyakovskiy and M'Hallah (2011)	1DBPP	x	x	min unused space + earliness + tardiness	y
Arbib and Marinelli (2014)	1DCSP	-	x	min stock lengths + tardiness	y
Gradišar and Glavan (2020)	VSBPP	-	x	min unused space + tardiness	y
Liu et al. (2021)	VSBPP	x	x	min cost	n
Arbib and Marinelli (2017)	1DBPP	-	x	min completion time + lateness	y

Reinertsen and Vossen presented an integer programming-based heuristic to minimize a combination of trim-loss and total weighted tardiness for solving the one-dimensional cutting stock problem (1DCSP) when orders must be completed before the due date [44]. In the model, the durations of a task are represented by items, and the due date is represented by the bin size available before the order expires. In addition to this, they assume that all due dates cannot be fully satisfied, which means that violating the due date constraint is allowed and

penalizes such delay violation in the objectives with a constant. Similarly, the delays incurred are represented by bin size. Another solution uses due dates as an additional constraint. Fazi, Van Woensel, and Fransoo extended the classical variable-size bin packing problem (VSBPP) by introducing a new variable that incorporates the time characteristics of bins and items [18]. This variable is random, i.e., the arrival time of an item is uncertain and follows a discrete probability distribution. Items can be processed at a given time and would be penalized for tardiness.

Others would add the penalty of early processing into consideration, such as Polyakovskiy and M'Hallah, who similarly argue that premature cutting would require temporary transfer and storage [42]. Therefore each item should be cut on the date before it enters the next stage of production (assembly or additional machining). Arbib and Marinelli developed an exact integer linear programming formulation with column generation [3]. This model contains a large number of variables associated with time-indexing, which are adjusted by an ad-hoc procedure. To solve the material requirements planning for a specific problem: a group of work orders for a product must be produced from the same batch of material, Gradišar and Glavan use integer planning to develop an extended bin packing problem formulation based on (i) variable case sizes, (ii) consideration of time constraints, and (iii) grouping of items/cases [20]. They claim that this approach can be applied to small and medium-sized problems. Liu et al. introduce novel algorithms for solving the variable size bin packing problem with time windows (VSBPTW) [32]. Lateness is considered infeasible in this paper, a bin must be loaded before expiration, and there must be a common time point between time windows of items within the same bin. Three heuristics are proposed to find the near-optimal solutions in a short time, namely the classical best-fit heuristic, the iterative local search based on the shortest path decoder, and the primal heuristic based on column generation.

The time-indexed formulation can be easily extended to different scheduling goals. However, a typical drawback of this formulation is the rather large number of variables and constraints [2]. Arbib and Marinelli proposed exact integer linear programming formulations and heuristic algorithms indexed by time for the one-dimensional bin packing problem with due dates, with the objective of minimizing a convex combination of the number of bins and the maximum delay of the items [2]. In their experience, the use of lower and upper bounds plays a crucial role in reinforcing the constraints, further strengthening the makespan and maximum lateness, and significantly reducing CPU time.

3.1.2. Two-dimensional Bin Packing Problem (2DBPP)

Later, numerous scholars solved the two-dimensional packing problem with due dates using integer programming and heuristics. The objective is to assign a set of n rectangular objects defined by width and height to the smallest number of identical rectangular boxes that are not overlapping [33]. The two-dimensional packing problems considering time factors of the previously cited papers are summarised in Table 3.2.

Bennell, Lee, and Potts extended the problem in subsection 3.1.1 to two dimensions. They used a genetic heuristic to solve the non-oriented, single bin-sized 2DBPP problem with a due date. The problem alternates periodically between the objectives of minimizing the maximum lateness and the number of bins used [6]. Marinelli and Pizzuti address the same problem by using a sequential value correction heuristic that further reduces the number of bins and the maximum delay time for 2DBPP with due dates[35]. Moreover, Polyakovskiy and M'Hallah explore the same problem described in Bennell, Lee, and Potts's work but focus on minimizing only the maximum delay time of the items [41]. For small instances, they use mixed integer programming (MIP) to model the problem while for larger instances the two-stage heuristic is

Table 3.2: Recent literature on 2DBPP addressing time constraints

Publications	Type	Orientation	Variable bin size	Availability	Due date	Objective
Bennell, Lee, and Potts (2013)	2DBPP	x	-	-	x	min lateness
Marinelli and Pizzuti (2018)	2DBPP	x	-	-	x	min bins + lateness
Polyakovskiy and M'Hallah (2018)	2DBPP	x	-	-	x	min lateness
Polyakovskiy and M'Hallah (2021)	2DBPP	-	-	x	x	min earliness + tardiness

used to approximate the solution. Polyakovskiy and M'Hallah's research uses integrated constraint program modeling to address both two-dimensional bin packing and just-in-time batch scheduling aspects of the problem [43]. They interact either cooperatively or competitively depending on the problem parameters (e.g., due date distribution, setup, and processing time), aiming not only to optimize space utilization in the bin packing process but also highlighting the importance of scheduling in operations. Just-in-Time (JIT) recognizes that there are costs associated with both missing the due date of an item and processing earlier than the due date.

While the methodology and results of 1DBPP and 2DBPP studies can provide some insights, their direct application to aircraft cargo loading is limited due to the specific practical challenges and operational needs of air cargo loading. 1D bin packing problems are primarily developed for cutting reinforcing steel bars, the 2D bin packing problem is mainly used for planar cuts such as paper and wood, and naturally does not consider vertical stacking constraints. In comparison, aircraft cargo in real life is a three-dimensional item with various shapes and sizes. This paper solves the 3D bin packing problem to determine the loading configuration for each ULD.

3.1.3. Three-dimensional Bin Packing Problem (3DBPP)

Due to the different shapes and sizes of air cargo pallets, the Bin packing problem can usually be regarded as a three-dimensional, non-identical multiple problem in air cargo loading [11]. We give an overview of the mentioned papers, their covered features, and objectives in Table 3.3. The columns display the features under consideration as well as the objective. Technical constraints such as non-overlapping and placement within a container, which appear in all articles, are not shown.

Table 3.3: Recent literature on application of 3DBPP in air cargo

Publications	Type	Weight limits	Weight distrib.	Orientation	Stacking	Stability	Availability	Due date	Objective
Chen, Lee, and Shen (1995)	3DBPP	-	x	x	-	-	-	-	min unused ULD volume
Lin, Chang, and Yang (2006)	3DBPP	x	-	x	x	x	-	-	min cost - load + effort
Chan et al. (2006)	3DBPP	x	x	x	-	x	-	-	1. min cost, 2. max load
Paquay, Schyns, and Limbourg (2016)	3DBPP	x	x	x	x	x	-	-	min unused ULD volume
Hai, Nananukul, et al. (2016)	3DBPP	x	-	-	-	-	x	-	min unused ULD volume
Paquay, Limbourg, and Schyns (2018)	3DBPP	x	x	x	x	x	-	-	min unused ULD volume
This work	**3DBPP**	**x**	**-**	**x**	**x**	**x**	**x**	**x**	**min unused ULD volume + delay**

In 1995, Chen, Lee, and Shen suggested a zero-one mixed-integer planning model to address the issue of loading containers with irregularly sized cartons [12]. Their model took into account various containers, carton sizes, orientation, and carton overlap within the container. Although the mathematical model gained the container's optimal total unused space, the enormous constraints and variables rendered the model computationally inappropriate for real-world applications. Lin, Chang, and Yang proposed a two-step solution divided into a grouping procedure and a packing procedure [31]: first, they used a K-means method to assign items across boxes according to different characteristics, such as the size of the object, the unloading order of the object and the capacity of the container. Then a genetic algorithm with heuristics is used to pack the items into each box according to the constraints. Chan et al. also proposed a two-stage approach [11]. The first stage applies a linear programming model to provide a lower bound on the global minimum cost of filling pallets with available products depending on their weight and volume. The second stage uses a heuristic to load cargo boxes and create a loading plan for each pallet. The optimal plan is reached when the outcome has the smallest variation from the lower bound determined in the first step. Crainic, Perboli, and Tadei proposed a new Extreme Point-based rule for loading items in three-dimensional containers, allowing better exploitation of bin volumes with a negligible amount of computational effort [16].

In the context of air cargo, Paquay, Schyns, and Limbourg proposed mixed-integer planning equations with detailed mathematical definitions, taking into account a wide range of constraints in the practical application of air cargo, as well as the specific case where the container is a truncated parallelepiped. However, this model is intended to be effective only for small-scale testing [40]. Paquay, Limbourg, and Schyns then developed a fast and constructive heuristic in two phases [39]: first, a packing algorithm is proposed for the same bin, and then multiple types of bins are processed in the second step, which has a low computation time: 12 seconds can be computed for loading 100 items.

In the 3DBPP for air cargo loading, researchers have developed various methods to deal with the complexity of the loading process, especially considering the various shapes and sizes of cargo pallets. However, although these methods have made some progress in spatial optimization, they usually do not directly take into account the time factor. Hai, Nananukul, et al. considered the time factor from a freight forwarder's point of view and proposed two models to perform daily air cargo loading management [22]. The first model is the air cargo 3D packing model, which loads cargo of multiple sizes and release dates into ULDs in order to minimize idle space in the ULDs. The second model, the air shipment assignment model, assigns each ULD to a specific airline with the goal of minimizing transportation costs while satisfying delivery constraints. These two models are solved sequentially and can complete the decision-making process for the freight forwarder to transport the cargo in a reasonable amount of time. In our paper, there is a difference in the perspective of solving the problem, since there is no case of assigning ULDs to different airlines. Although the focus of this paper is on the former model, there is a difference in the introduction of the concept of time in the development of the mathematical model, which will take into account the item issue time, deadline, and packing processing time.

3.2. Build-up Scheduling Problem

The problem of build-up scheduling has many characteristics in common with the Parallel Machine Scheduling Problem (PMSP). The PMSP involves scheduling a set of tasks on two or more machines. Not only is it necessary to find the optimal job ordering, but also the optimal assignment of job machines [38]. Parallel job machines include three categories [8]:

- Identical machines: all machines process the same job at the same rate.
- Uniform machines: each machine executes all jobs at the same speed.
- Unrelated machine: there is no specific relationship between job and machine processing speed.

The previously literatures are summarised in Table 3.4.

Table 3.4: Recent literature on PMSP

Publications	Type	Objective
Blazewicz, Dror, and Weglarz (1991)	Identical, uniform, and unrelated PMSP	min max completion time/makespan
Kan (1979)	Uniform PMSP	min cost
Chen and Powell (1999)	Identical, non-identical, and unrelated PMSP	min total weighted completion time and weighted number of tardy jobs
Chen and Powell (1999)	Identical PMSP	min total weighted earliness and tardiness
Chung-Yee Lee(2000)	Identical PMSP	min total weighted completion time
Bank and Werner (2001)	Unrelated PMSP	min total weighted earliness and tardiness cost
Van Den Akker, Hoogeveen, and Kempen (2006)	Identical PMSP	min max lateness
Unlu and Mason (2010)	Identical PMSP	min total weighted completion time/makespan /maximum lateness/total weighted tardiness and total number of tardy jobs
Li, Sivakumar, and Ganesan (2008)	Unrelated PMSP	min total weighted earliness
Arık (2020)	Unrelated PMSP	min total weighted earliness and tardiness cost
This work	**Unrelated PMSP**	**min total completion time**

First, Blazewicz, Dror, and Weglarz first laid out a large number of mathematical formulations for the PMSP and provided several variants of it, including Non-preemptive scheduling on identical parallel machines and Parallel uniform processors with unit standard processing times [8]. Kan formulated the latter problem as a special case of the simple traffic network problem and provided an efficient solution [27]. Chen and Powell proposed a decomposition methodology applied to the implementation of identical, non-identical, and non-related PMSPs [15]. And they obtained computational results with the objective of minimizing the total weighted completion time and the weighted number of tardy jobs. Chen and Powell also proposed an exact decomposition algorithm based on column generation in the same year, which is used to minimize the total weighted advancement and tardiness of the identical parallel machine problem [14]. And the algorithm is capable of solving problems with up to 60 jobs in a reasonable amount of CPU time. Furthermore, Van Den Akker, Hoogeveen, and Kempen applied the column generation technique to the min-max objective function and succeeded in analyzing the objective of minimizing the maximum lateness of the problem in a reasonable time for a larger number of jobs and identical parallel machine examples [46].

For the unrelated PMSP, Bank and Werner proposed various constructive and iterative heuristics whose goal is to assign jobs with pre-determined common due dates to machines with different release dates and to schedule the jobs assigned to each machine so that the weighted sum of the early and late penalties is minimized [5]. Furthermore, Unlu and Mason proposed four different MIP formulas based on four different types of decision variables involving total weighted completion time, completion time, maximum delay, total weighted delay, and total delay for various PMSPs [45]. And the efficiency of the different formulas is tested to get the promising paradigm of optimization formulas. In simultaneous scheduling problem for air transport, Li, Sivakumar, and Ganesan formulated the PWSP problem with earliness/tardiness penalties using MIP to minimize the total earliness of all operations of the assembly to ensure that orders are completed on time and in time for the flights [29]. In addition to minimizing the

earliness, Arık also considered the tardiness cost into the objective function to minimize the total weighted sum of earliness and tardiness costs [4]. He used three different meta-heuristics to solve the unrelated PMSP and found that the group intelligence-based meta-heuristic can be the most efficient. Alessandro et al.'s publication delves deeply into the PMSP, emphasizing several variations of how a group of tasks might be planned on two or more machines [1]. The variant they suggest requires that a task be processed entirely on one machine, and it is not permitted to get started a task on one machine and then transfer it to another to complete. In their example, this mathematical model assists students in assigning exam study activities while taking into account the essential prior conditions to fulfill them as soon as possible. This case vividly illustrates how PMSP may be used for real-world learning and time management, as well as insights into the air cargo scheduling challenge. There is also a PMSP variant that assumes that machines are not always available, since in most real-world situations, machines need to be maintained at certain periods. Lee and Chen then investigated both cases where multiple machines can be maintained at the same time and where only one machine can be maintained at a given time for this possibility [28]. They used a column generation-based algorithm to solve the work and maintenance activities to minimize the total weighted completion time of the work.

In the preceding literature review, we note that PMSP has been extensively studied, mainly involving different types of machines (identical, uniform, and unrelated machines) and multiple scheduling strategies. Similar to some of the studies in the traditional literature, this paper highlights minimizing the final completion time of unrelated parallel machine scheduling as the key objective. Unlike these studies, however, we focus specifically on the time efficiency of ULD construction in air freight applications, where the build-up task is constrained by strict time windows. These time windows are defined by the maximum release date and minimum due date of the items within the ULD to ensure that each ULD is built no earlier than the item's release date and must be completed before the deadline.

3.3. Discussion

The above sections summarise the literature related to APP and BSP. Firstly, a variety of approaches to the bin packing problem, spanning from one-dimensional to three-dimensional contexts, have been encapsulated for APP. It is evident from the literature that the evolution of the problem has transitioned from simple, one-dimensional scenarios to more complex and practical applications such as air cargo loading, which require a three-dimensional approach. The focus on time constraints, in particular, highlights the problem's practical significance and leads the development of complicated models that can manage the complexities of real-world situations.

The reviewed works demonstrate that while advancements have been achieved in integrating time-related constraints into one- and two-dimensional bin packing problems, there are still issues to be resolved, especially in adapting these models to the intricate requirements of air cargo loading. The existing literature on 3DBPP primarily focuses on optimizing spatial utilization, often sidelining temporal factors that play a significant role in real-world applications. Future research should strive to bridge this gap by constructing integrated models that account for the time sensitivity of air cargo, incorporating elements such as cargo availability, ULD readiness, and synchronization with the flight schedule. There is also the opportunity to integrate real-time data and predictive analytics, allowing for dynamic adjustments to packing strategies based on logistical considerations. Furthermore, developing scalable and computationally feasible models is critical to ensure that solutions are both theoretically sound and practicable for real-world applications.

In terms of the BSP, the traditional literature on PMSP has yet to address the completion of ULDs within a tight time window specific to the air cargo context. This is in contrast to Mokotoff and Blazewicz, Dror, and Weglarz, who, while discussing in detail different configurations and scheduling strategies for parallel machines, often fail to adequately take into account the time constraints in industry-specific applications. Also different from the rest of the literature where earliness and lateness are allowed but penalized. Since the flight departure time is fixed, a delay in the build-up will cause the ULDs not to be shipped on time. Therefore, future research on the build-up scheduling problem in the air cargo context should consider a strict time window for each job to ensure that the build-up of each ULD can be completed on time.

In conclusion, future research should endeavor to develop comprehensive models for the air cargo build-up process. First, time sensitivity should be incorporated into the spatial optimization problem for air cargo, and second, the strict time window for ULD construction should be considered in the BSP. Thus, the subtle demands of air cargo logistics should be captured, enabling solutions that are both spatially and temporally optimum.

4

Mathematical formulation and Algorithm

4.1. 3DBPP-Mixed Integer Programming

In this section, a MIP model for the 3DBPP in the context of air cargo is introduced, and the following problem description is given. A set of n rectangular boxes with dimensions $l_i \times w_i \times h_i$ and weights $weight_i$ ($i \in 1, ..., n$) of n rectangular boxes with dimensions $L_j \times W_j \times H_j$ and a maximum capacity (also called maximum gross weight) of max_weight_j into m available ULDs while minimizing the unused volume. The model also takes into account the typical commitment of a cross-border e-commerce company to its customers, i.e., to ensure that the cargo is loaded and then shipped by air within a specified due date. The packaging must satisfy different geometric and time-dependent constraints that will be specified later.

4.1.1. Sets

N	Set of items to be loaded,
M	Set of available ULDs.

4.1.2. Parameters

The following inputs are necessary before solving 3DBPP with due dates.

Items

- *Dimensions and Weight of Items:* The physical attributes of each cargo item, including its height, width, depth, and weight.
- *Item Release Time and Deadline:* The moment an order is placed and the deadline for the cargo to be packed and ready for handover to the airline.
- *Penalty Coefficient for Delayed Deliveries:* A numerical value representing the penalty imposed for cargo delivered past its deadline.
- *Maximum Delay Allowed:* The maximum time each item can be delayed for loading ensures that the delays are all manageable.

ULDS

- *Dimensions and Available Quantity of ULDs:* The size specifications of the Unit Load Devices (ULDs) and the number available for cargo loading.

- *Cost of ULDs:* The expense associated with using Unit Load Devices for cargo transportation.
- *ULD Packing Processing Time:* The amount of time required to pack and prepare a ULD for shipping.

Based on the above-known data, the parameters of the model are referred to as:

l_i, w_i, h_i	Length, width, and height of item i,	$\forall i \in N$,
$weight_i$	Weight of item i,	$\forall i \in N$,
v_i	Volume of item i,	$\forall i \in N$,
R_i	Release date of item i,	$\forall i \in N$,
DDL_i	Due date of item i,	$\forall i \in N$,
L_j, W_j, H_j	Length, width, and height of ULD j,	$\forall j \in M$,
max_weight_j	Maximum gross weight of ULD j,	$\forall j \in M$,
V_j	Volume of ULD j,	$\forall j \in M$,
$cost_j$	Cost per kg of using the ULD j,	$\forall j \in M$,
P	Penalty per unit of time delay,	
max_delay	Maximum delay allowed per item,	
Tp	Average processing time per item per ULD loading,	
L	Maximum length for all ULD types,	
W	Maximum width for all ULD types,	
H	Maximum height for all ULD types.	

4.1.3. Variables

Here are the various variables used in the model.

Geometric variables

$$n_{ij} = \begin{cases} 1 & \text{if item } i \text{ is placed in ULD } j, \\ 0 & \text{otherwise}, \end{cases} \quad \forall i \in N, j \in M,$$

$$u_j = \begin{cases} 1 & \text{if ULD } j \text{ is used}, \\ 0 & \text{otherwise}, \end{cases} \quad \forall j \in M,$$

(x_i, y_i, z_i) Location of the front left bottom corner of item i, $\quad \forall i \in N$,

(x'_i, y'_i, z'_i) Location of the rear right top corner of item i, $\quad \forall i \in N$,

$$r_{iab} = \begin{cases} 1 & \text{if the side } b \text{ of item } i \text{ is along the } a\text{-axis}, \\ 0 & \text{otherwise}, \end{cases} \quad \forall i \in N,$$

$$r_{ik} = \begin{cases} 1 & \text{if item } i \text{ is on the right of item } k (x'_k \leq x_i), \\ 0 & \text{otherwise } (x'_k > x_i), \end{cases} \quad \forall i \in N,$$

$$b_{ik} = \begin{cases} 1 & \text{if box } i \text{ is behind box } k (y'_k \leq y_i), \\ 0 & \text{otherwise } (y'_k > y_i), \end{cases} \quad \forall i \in N,$$

$$a_{ik} = \begin{cases} 1 & \text{if item } i \text{ is above item } k (z'_k \leq z_i), \\ 0 & \text{otherwise } (z'_k > z_i), \end{cases} \quad \forall i \in N.$$

Specific variables

As mentioned already, applying the 3DBPP with due date to the real world situations implies some specific variables.

- Vertical stability

$$g_i = \begin{cases} 1 & \text{if item } i \text{ is on the ground } (z_i = 0), \\ 0 & \text{otherwise}, \end{cases} \quad \forall i \in N,$$

$$h_{ik} = \begin{cases} 0 & \text{if item } k \text{ has the suitable height to support item } i (z_i = z'_k), \\ 1 & \text{otherwise}, \end{cases} \quad \forall i, k \in N,$$

$$o_{ik} = \begin{cases} 0 & \text{if the item } i \text{ and item } j \text{ overlap on the } XY \text{ plane}, \\ 1 & \text{otherwise}, \end{cases} \quad \forall i, k \in N,$$

$$s_{ik} = \begin{cases} 1 & \text{if item } k \text{ supports item } i \text{ and are in the same ULD}, \\ 0 & \text{otherwise}, \end{cases} \quad \forall i, k \in N,$$

$$\eta^1_{ik} = \begin{cases} 0 & \text{if } x_k \leq x_i, \\ 1 & \text{otherwise}, \end{cases} \quad \forall i, k \in N,$$

$$\eta^2_{ik} = \begin{cases} 0 & \text{if } y_k \leq y_i, \\ 1 & \text{otherwise}, \end{cases} \quad \forall i, k \in N,$$

$$\eta^3_{ik} = \begin{cases} 0 & \text{if } x'_i \leq x'_k, \\ 1 & \text{otherwise}, \end{cases} \quad \forall i, k \in N,$$

$$\eta^4_{ik} = \begin{cases} 0 & \text{if } y'_i \leq y'_k, \\ 1 & \text{otherwise}, \end{cases} \quad \forall i, k \in N,$$

$$\beta^l_{ik} = \begin{cases} 1 & \text{if the vertex } l \text{ is supported by item } k, \\ 0 & \text{otherwise}, \end{cases} \quad \forall i, k \in N, l \in \{1, ..., 4\}.$$

- Time-related

d_i Delay time of item i, $\forall i \in N$,

s_j Earliest start processing time of ULD j, $\forall j \in M$,

end_j Earliest end processing time of ULD j, $\forall j \in M$,

r_j The minimum time allowed in ULD j to start loading items, which is the latest release time of all loaded items, $\forall j \in M$,

min_ddl_j The maximum due date of the loaded item in ULD j, i.e. the earliest due date of all loaded items, $\forall j \in M$.

4.1.4. Objective function

The objective is to maximise ULD utilisation whilst trying to meet delivery dates. That is, to minimize the delay of cargo relative to its DDL while using the least costly ULDs possible. The objective function is to minimise the unused volume of the ULDs and delay penalties.

$$Min \sum_{j \in M} u_j \cdot V_j - \sum_{i \in N} v_i + P \cdot \sum_{j \in M} \sum_{i \in N} d_i \cdot n_{ij} \tag{4.1}$$

Above objective function reflects a trade-off between achieving ideal packing, which minimizes waste but may result in some cargo being delayed past its DDL—potentially leading to customer churn or compensation for delays—and packaging goods strictly according to the DDL, which may increase ULD usage costs.

Since v_i acts as a starting parameter, the expression $\sum_{i \in N} v_i$ is a constant. Therefore, the volume of the used containers is minimized:

$$Min \sum_{j \in M} u_j \cdot V_j \tag{4.2}$$

Moreover, before describing the constraints in detail, it is important to first linearise the product of a continuous variable d_i with a binary variable n_{ij}. A continuous variable z_{ij} is introduced to hold the product: $z_{ij} = d_i \cdot n_{ij}, \quad \forall i, j$. Thus the final objective function is:

$$Min \sum_{j \in M} u_j \cdot V_j + P \cdot \sum_{j \in M} \sum_{i \in N} z_{ij} \tag{4.3}$$

4.1.5. Constraints

Before going into any more detail,the following constraints 4.4-4.6 need to be added to force z_{ij} to take on $d_i \cdot n_{ij}$ values, where n_{ij} is a binary variable and d_i is a continuous variable for which $0 \leq d_i \leq max\_delay$ hold.

$$z_{ij} \leq max\_delay \cdot n_{ij}, \quad \forall i \in N, j \in M \tag{4.4}$$

$$z_{ij} \leq d_i, \quad \forall i \in N, j \in M \tag{4.5}$$

$$z_{ij} \geq d_i - max\_delay \cdot (1 - n_{ij}), \quad \forall i \in N, j \in M \tag{4.6}$$

Geometric constraints

$$\sum_{j \in M} n_{ij} = 1, \quad \forall i \in N \tag{4.7}$$

$$n_{ij} \leq u_j, \quad \forall i \in N \tag{4.8}$$

The constraint 4.7 requires that all items should be allocated in ULDs, and constraint 4.8 marks the ULD as used, once an item has been placed into it [7][40].

$$\sum_{i \in N} weight_i \cdot n_{ij} \leq max\_weight_j \cdot u_j, \quad \forall j \in M \tag{4.9}$$

The maximum capacity of each ULD cannot be exceeded. There is a maximum allowable weight for loaded items for each ULD j [7][11][40], which is ensured by constraint 4.9.

$$x'_i \leq \sum_{j \in M} L_j n_{ij}, \quad \forall i \in N \tag{4.10}$$

$$y'_i \leq \sum_{j \in M} W_j n_{ij}, \quad \forall i \in N \tag{4.11}$$

$$z'_i \leq \sum_{j \in M} H_j n_{ij}, \quad \forall i \in N \tag{4.12}$$

All cargo must be completely within the permitted contour of the assigned ULD [40]. The constraints 4.10, 4.11, and 4.12 ensure that the all the items do not exceed their ULD size.

$$x'_i - x_i = r_{i11} l_i + r_{i12} w_i + r_{i13} h_i, \quad \forall i \in N \tag{4.13}$$

$$y'_i - y_i = r_{i21} l_i + r_{i22} w_i + r_{i23} h_i, \quad \forall i \in N \tag{4.14}$$

$$z'_i - z_i = r_{i31} l_i + r_{i32} w_i + r_{i33} h_i, \quad \forall i \in N \tag{4.15}$$

$$\sum_{a} r_{iab} = 1, \quad \forall i \in N, b \in \{1, 2, 3\} \tag{4.16}$$

$$\sum_{b} r_{iab} = 1, \quad \forall i \in N, a \in \{1, 2, 3\} \tag{4.17}$$

Items must be placed parallel to the ULD, with their edges aligned with the corresponding edges of the ULD [40]. Thus items can rotate orthogonally [12][7][40], and the variables r_iab are introduced to describe the orientation of the item i inside the ULD. Under such definition, Constraint 4.13, 4.14, and 4.15 are applied to define the position of the rear right top corner of the item according to its placement direction. Together with constraints 4.16 and 4.17, they describe that items can be rotated orthogonally in the ULD.

$$r_{ik} + r_{ki} + b_{ik} + b_{ki} + a_{ik} + a_{ki} \geq (n_{ij} + n_{kj}) - 1, \quad \forall i, k \in N, j \in M \tag{4.18}$$

No point within any two pieces of cargo can occupy the same position in space [40]. Therefore at least one axis is not allowed to overlap, i.e., at least one of these variables on the left-hand side of constraint 4.18 must be equal to 1. In addition to this, overlap is only constrained if two items are located in the same ULD, which is indicated by the right-hand side of constraint 4.18.

$$x'_k \leq x_i + (1 - r_{ik}) \cdot L, \quad \forall i, k \in N \tag{4.19}$$

$$x_i + 1 \leq x'_k + r_{ik} \cdot L, \quad \forall i, k \in N \tag{4.20}$$

$$y'_k \leq y_i + (1 - b_{ik}) \cdot W, \quad \forall i, k \in N \tag{4.21}$$

$$y_i + 1 \leq y'_k + b_{ik} \cdot W, \quad \forall i, k \in N \tag{4.22}$$

$$z'_k \leq z_i + (1 - a_{ik}) \cdot H, \quad \forall i, k \in N \tag{4.23}$$

Constraints 4.19-4.23 give declarations to variables r_{ij}, b_{ij}, and aij, which defines the relative location of two items and links to the restrictions on whether two items are actually placed in the same ULD with overlap constraint 4.18. Moreover, the parameters L, W and H are used in these constraints because it is not known in which ULD i and k are located.

Vertical stability constraints

Based on the variables introduced in section 4.1.3, stability constraints could be written as follows.

$$\sum_{l=1}^{4} \sum_{k \in N} \beta_{ik}^{l} \geq 3(1 - g_i), \quad \forall i \in N \tag{4.24}$$

Items cannot be stacked without support. There must be other items or the base of the container beneath them. Thus the constraint 4.24 determines if the item i is on the ground. If not, this constraint ensures that at least three vertices are supported by one or more items k.

$$z_i \leq (1 - g_i)H, \quad \forall i \in N \tag{4.25}$$

According to the constraint 4.25, if item i is on the ground ($g_i = 1$), then the value of the z-coordinate of the front left bottom corner of item i must be zero.

$$z'_k - z_i \leq v_{ik}, \quad \forall i, k \in N \tag{4.26}$$

$$z_i - z'_k \leq v_{ik}, \quad \forall i, k \in N \tag{4.27}$$

$$v_{ik} \leq z'_k - z_i + 2H(1 - m_{ik}), \quad \forall i, k \in N \tag{4.28}$$

$$v_{ik} \leq z_i - z'_k + 2Hm_{ik}, \quad \forall i, k \in N \tag{4.29}$$

$$h_{ik} \leq v_{ik}, \quad \forall i, k \in N \tag{4.30}$$

$$v_{ik} \leq h_{ik}H, \quad \forall i, k \in N \tag{4.31}$$

In the constraints 4.26-4.31, two intermediate variables are used to define the variables h_{ij}, v_{ik} and m_{ik}. v_{ik} is used as a representation of the absolute value of $|zk' - zi|$, while m_{ik} is a binary variable with a value of 1 when z'_k is greater than or equal to z_i, and 0 otherwise.

$$o_{ik} \leq r_{ik} + r_{ki} + b_{ik} + b_{ik} \leq 2o_{ik}, \quad \forall i,k \in N \tag{4.32}$$

$$(1 - s_{ik}) \leq h_{ik} + o_{ik} \leq 2(1 - s_{ik}), \quad \forall i,k \in N \tag{4.33}$$

Next, constraints 4.32 and 4.33 are added to make a full determination of the overlapping variable o_{ik}. If o_{ik} equals to 0, item i and k share a part of their orthogonal projection on the XY plane. The latter indicates that $h_{ik} + o_{ik} = 0$ if the bottom surface of item i is supported by the top surface of item k.

$$n_{ij} - n_{kj} \leq 1 - s_{ik}, \quad \forall i,k \in N, j \in M \tag{4.34}$$

$$n_{kj} - n_{ij} \leq 1 - s_{ik}, \quad \forall i,k \in N, j \in M \tag{4.35}$$

$$\beta^l_{ik} \leq s_{ik}, \quad \forall i,k \in N, l \in \{1,...,4\} \tag{4.36}$$

It is also necessary to fully identify the variable s_{ik}. Constraints 4.34 and 4.35 ensure that item i is supported by item k only if these two items are in the same ULD j. Secondly, one of vertex of item i can be supported by item k only if item i is supported by k, which is guaranteed by constraint 4.36.

$$\eta^1_{ik} + \eta^2_{ik} \leq 2(1 - \beta^1_{ik}), \quad \forall i,k \in N \tag{4.37}$$

$$\eta^2_{ik} + \eta^3_{ik} \leq 2(1 - \beta^2_{ik}), \quad \forall i,k \in N \tag{4.38}$$

$$\eta^3_{ik} + \eta^4_{ik} \leq 2(1 - \beta^3_{ik}), \quad \forall i,k \in N \tag{4.39}$$

$$\eta^1_{ik} + \eta^4_{ik} \leq 2(1 - \beta^4_{ik}), \quad \forall i,k \in N \tag{4.40}$$

In constraints 4.37-4.40, η^1_{ik} and η^2_{ik} represent the relative position of the front left vertices ($l = 1$) of item i and item k, η^2_{ik} and η^3_{ik} refer to the relative position of the front right vertices ($l = 2$), η^3_{ik} and η^4_{ik} indicate the rear right vertices ($l = 3$), and η^4_{ik} denote the rear left vertices ($l = 4$). The relative positions of the two items at each of the four vertices are used to define the case where the four corners of the basis of item i are supported by item k.

$$x_k \leq x_i + \eta^1_{ik} \cdot L, \quad \forall i,k \in N \tag{4.41}$$

$$y_k \leq y_i + \eta^2_{ik} \cdot W, \quad \forall i,k \in N \tag{4.42}$$

$$x'_i \leq x'_k + \eta^3_{ik} \cdot L, \quad \forall i,k \in N \tag{4.43}$$

$$y'_i \leq y'_k + \eta^4_{ik} \cdot W, \quad \forall i,k \in N \tag{4.44}$$

The constraints 4.41-4.44 further define four ηs. Constraints 4.41 and 4.43 ensure that η^1_{ik} and η^3_{ik} are equal to 0 if $x_i \geq x_k$ and $x'_i \leq x'_k$. Likewise, constraints 4.42 and 4.44 define the η^2_{ik} and η^4_{ik} on the y-axis.

Time-related constraints

$$s_j \geq R_i \cdot n_{ij}, \quad \forall i \in N, j \in M \tag{4.45}$$

Items can only be loaded onto a ULD after the order has been issued. Therefore, the actual start processing time (s_j) of ULD j is equal to or greater than the release date of all items in it. The constraint 4.45 is written to define the maximum release time for each ULD, limited by the items loaded in it.

$$end_j \geq s_j + Tp \cdot \sum_{i \in N} n_{ij}, \quad \forall i \in N, j \in M \tag{4.46}$$

Items also need to be scheduled on the ULD before the deadline. The end_j of ULD j is defined by constraint 4.46: it is equal to the sum of its start time s_j and total processing time $t_p \cdot \sum_{i=1}^{n} n_{ij}$.

$$d_i \geq \sum_{j \in M} end_j \cdot n_{ij} - DDL_i, \quad \forall i \in N \tag{4.47}$$

Constraint 4.47 defines the delay d_i for each item. It is 0 if there is no delay, otherwise it is the difference between the end loading time of the ULD (end_j) where item i is located and the due date of item i.

$$0 \leq d_i \leq max\_delay, \quad \forall i \in N \tag{4.48}$$

Constraint 4.48 sets the delay time d_i for each item that must be within the maximum threshold max_delay.

4.2. 3DBPP-Extreme Point Heuristic Algorithm

As the problem's size grows, so does the complexity of addressing the 3DBPP. To address this issue, this chapter will propose heuristic algorithms. The problem statement is identical to that of the previous chapter: pack all things into the least amount of ULDs possible while avoiding overlapping and ensuring stable placement. The algorithm will use space effectively while monitoring the loading completion time, attempting to remain ahead of the due date. One additional assumption is added that there is no limit to the number of workstations at this stage and that the packing can be done synchronously no matter how many ULDs there are. The parallel machine scheduling problem is addressed in the next section.

The EPH is able to quickly generate a better feasible solution and avoid the computational overhead of a large-scale search by preferentially selecting the extreme points (i.e., the possible placement points of the items) inside the container as the placement points for the next item. The procedure of the algorithm is outlined below, with each step discussed in detail later.

1. **Initialization:**
    - **Item Sorting:** Sort all items to be loaded according to the priority rule.
    - **Container Sorting:** Sort the containers by capacity from largest to smallest, ensuring that larger containers are prioritized during the algorithm's execution to minimize the total number of containers used.
2. **Extreme Point Selection:** For each item to be placed, traverse the current set of extreme points. Check whether the item can be placed at each extreme point in all possible orientations without overlapping with already placed items, and ensure vertical stability.
3. **Item Placement:** Each item is loaded onto the best extreme point of the existing ULD using the best-fit decreasing (BFD) algorithm; if this is not possible, placement in a new ULD is attempted.
4. **Updating the Set of Extreme Points:** After an item is placed, update the set of extreme points in the ULD.
    - **Add new extreme points**
    - **Delete duplicate extreme points**
5. **Iteration:** Repeat steps 2 to 4 until all items are placed or no suitable extreme points are available for the remaining items.
6. **Result Output:** Output the placement of items in all ULDs, as well as the start and finish times of loading.

4.2.1. Sorting the items

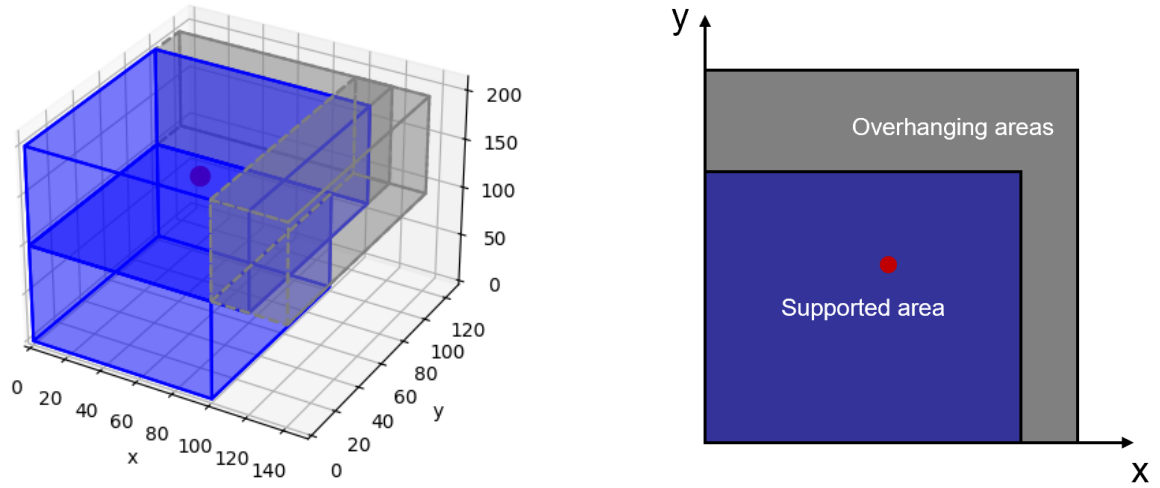
Different variants of the EP-BFD heuristic algorithm can be defined by modifying the order of the components. Based on the context of this paper's time-dependent factors, we used the following ranking principle.

Release Date - Due Date - Height: Items are sorted by the non-increasing value of their release date (R_i), whereas items with the same date are sorted by the non-increasing value of their due date (DDL_i). If the dates remain the same, the items are sorted by the non-increasing value of their height, h_i.

4.2.2. Ensuring vertical stability

To maintain the vertical stability of stacked items in a 3DBPP, it is important to ensure that adequate bottom support is provided for each item. A crucial constraint is included: each box stacked on top of another has no more than a particular proportion of its base area unsupported (i.e., overhanging). This virtually guarantees that items will not overturn during handling and transport. Usually, some studies allow the user to customize the contact area threshold, which can be adjusted to suit the crating scenario and specific needs [25][47]. To build a stable pallet, this specific ratio is generally between 70% [21] and 100% [13].

Specifically, suppose a box is put over another box that is smaller in the horizontal plane (XY plane). In that case, the top box's overhanging area shall not exceed a certain proportion of the total base area. This requirement does not apply to boxes placed directly on the ground or to boxes of equal or greater size in the XY plane, as these circumstances are intrinsically stable.



(a) 3D diagram

(b) 2D X-Y diagram

Figure 4.1: Example for stable support

Consider a scenario where a smaller box with dimensions 100 cm × 100 cm × 100 cm is placed underneath a larger box with dimensions 140 cm × 120 cm × 100 cm. As depicted in the Figure 4.1, the larger box extends beyond the edges of the smaller box, creating an overhang. The overhanging areas are highlighted with dashed lines and shaded in grey. The figure also marks the center of gravity of the box above (see red dot). The projection of the overhang onto the XY plane is calculated to ensure that it does not exceed the set percentage of the larger box's base area. By adhering to this constraint, the stacking process ensures that the top box remains stably supported, preventing potential tipping or imbalance in the packed configuration.

4.2.3. Searching for Extreme Points

The process of finding the Extreme Points (EPs) is based on the locations of items already placed in the container. Whenever a new item is placed in a container, certain points of that item are projected onto orthogonal coordinate axes to generate new potential placement locations, i.e., new EPs, as shown in Figure 4.2. The generation of new EPs is divided into the following two cases:

Empty container case

If no items have been placed in the container yet, the first item k will be placed in the lower above left corner of the container (at position $(0, 0, 0)$). The generated EPs are in positions:

- $(l_k, 0, 0)$ along the X axis,
- $(0, w_k, 0)$ along the Y axis,
- $(0, 0, h_k)$ along the Z axis,

Existing items in container

The item is placed at (x_k, y_k, z_k) and new EPs will be acquired:

- In the direction of the Y- and Z-axes: A new EP is generated from the projection position of the corner point $(x_k + l_k, y_k, z_k)$ of item k in the Y and Z axes directions.

- In the direction of the X- and Z-axes: A new EP is generated from the projection position of the corner point $(x_k, y_k + w_k, z_k)$ of item k in the X and Z axes directions.
- In the direction of the X- and Y-axes: A new EP is generated from the projection position of the corner point $(x_k, y_k, z_k + h_k)$ of item k in the X and Y axes directions.

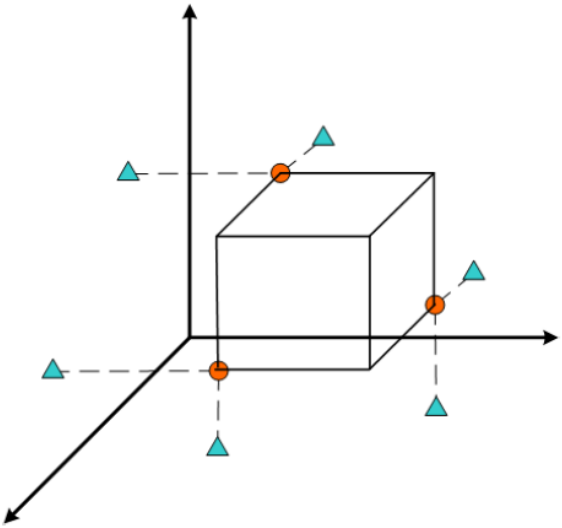


Figure 4.2: EPs defined by an item, represented by triangles [16]

If a newly generated EP overlaps with more than one item during projection, the point whose corner point is projected onto the item closest to k and the container wall is the final EP position. This processing guarantees that the created EPs precisely fit the already placed items, thus maximizing the use of space. After generating new EPs, the model adds these EPs to the EP list. To ensure the validity of the EP list, all newly generated poles are sorted in z, y, x order, and duplicates are removed. This step ensures that each EP is unique and can be efficiently utilized for subsequent item placement.

4.2.4. Best Fit Decreasing procedures

The Best-Fit Decreasing (BFD) algorithm attempts to place each item onto the EP of the existing bins, where the EP is evaluated by a merit function that maximizes the measurement of the best bin and the optimal placement position for the item. Specifically, from all feasible EPs and directions, the algorithm selects the placement that yields the best merit function value. If an item cannot fit into any of the current bins, a new ULD is created [37].

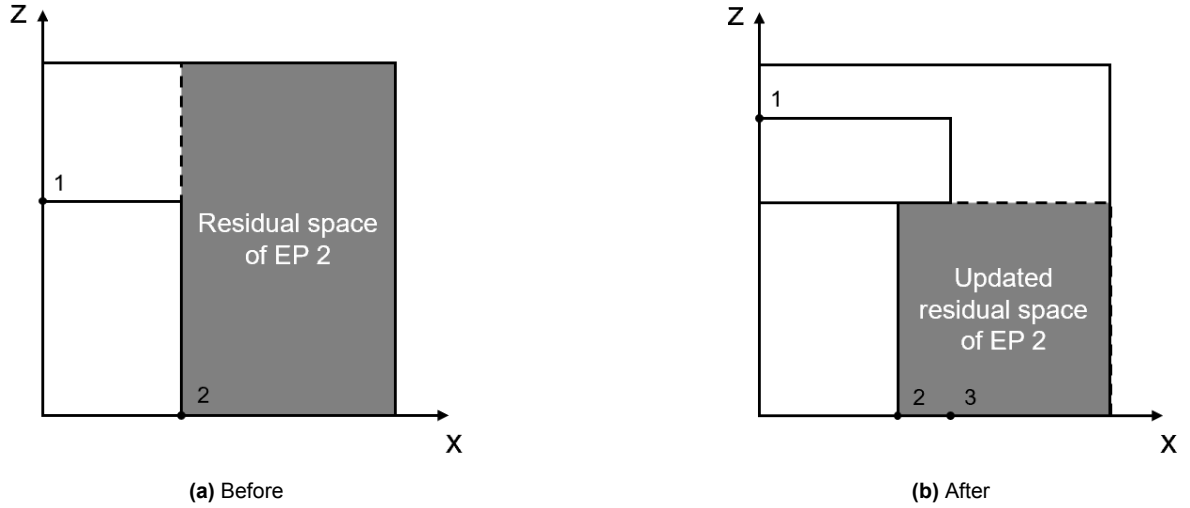
The merit function utilized in this paper is based on maximizing the utilization of the Residual Space (RS) of the EPs. RS measures the available free space around an EP, defined as the distance from the EP to the edge of the bin or the nearest item. This distance may vary along each axis. Therefore, when an EP is created, its RS on each axis is set to the distance from its position to the nearest obstruction along that axis. Every time a new item is added to the packing, the RS of all EPs is updated by the algorithm.

The RS update algorithm checks each EP in the EP list generated in the previous section. If an EP is within the range of the newly placed item, the algorithm calculates the RS for the relevant axes. Specifically:

For the x-axis
If the EP is within the z-range of the new item and on the side in the y-direction, the RS_x is updated to be the minimum between the current RS_x and the distance from the new item's position to the EP.

For the y-axis
Similarly, if the EP is within the z-range and on the side in the x-direction, the RS_y is updated to the minimum between the current RS_y and the distance from the EP to the new item.

For the z-axis
If the EP is below the new item and on its XY-plane, RS_z is updated to the minimum between the current RS_z and the distance from the EP to the new item's bottom. As shown in Figure 4.3, the RS of EP2 is then updated due to the new item being placed above it.



(a) Before **(b)** After

Figure 4.3: Example of RS definition in X and Z directions

These updates ensure that the RS reflects the current state of available space around each EP, leading to more precise and efficient space utilization. To more precisely determine the placement position, the merit function in this paper minimizes the difference between the RS of the EP and the dimensions of the item, defined as:

$$f_b = \left[(RS_e^x - l_k) + (RS_e^y - w_k) + (RS_e^z - h_k) \right],$$

where RS_e^x, RS_e^y, and RS_e^z represent the RS on the X, Y, and Z axes, respectively. In this way, the algorithm selects the EP and direction that minimize the difference between the remaining space and the item's dimensions from all feasible EPs and directions, thereby maximizing space utilization and ensuring the efficiency and effectiveness of the packing process.

4.3. Build-up Scheduling Problem

This section introduces the second challenge in the build-up loading process in the context of air freight, the Build-up Scheduling Problem. It is often known as the Parallel Machine Scheduling Problem (PMSP), in which jobs are handled across numerous machines. Each job has a set processing time, release time, and deadline. The mathematical model described below describes how jobs are assigned and sequenced on machines so as to minimize the maximum completion time of all jobs.

4.3.1. Sets

N Set of jobs that need to be scheduled,
M Set of available machines.

4.3.2. Parameters

The following inputs are necessary before solving the above PMSP.

ULDs

- *ULD Release Time and Deadline:* The time allowed to begin construction of the ULD and the deadline for completion of ULD handover to the airline.
- *ULD Packing Processing Time:* The time required to pack and prepare a ULD for shipping.

Workstation

- *Workstation Availability:* The number of available workstations

Based on the above known data, the parameters of the model are referred t as:

P_i Processing time of job i, $\forall i \in N$,
R_i Release date of job i, $\forall i \in N$,
D_i Deadline by which job i must be completed. $\forall i \in N$.

4.3.3. Variables

$$x_{ij} = \begin{cases} 1 & \text{if job } i \text{ is assigned to machine } j, \\ 0 & \text{otherwise,} \end{cases} \quad \forall i \in N, j \in M,$$

$$y_{ik} = \begin{cases} 1 & \text{if job } i \text{ precedes job } k, \\ 0 & \text{otherwise,} \end{cases} \quad \forall i, k \in N,$$

s_i start time of job i, $\forall i \in N$,

c_i completion time of job i, $\forall i \in N$,

c the latest completion time across all jobs.

4.3.4. Objective function

The quality of the schedule is measured by some objective function of minimizing the maximum completion time, that is, the makespan of the schedule. Therefore, the primary objective of this problem is to minimize the makespan, which is the total time required to complete all ULDs. This ensures that the cargo terminal operates efficiently and that flights can be dispatched as scheduled. The objective function is represented specifically as:

$$Min\ c \tag{4.49}$$

4.3.5. Constraints

In addition, job prioritization, non-overlapping constraints, and machine allocation are essential to ensure feasible and optimal scheduling.

$$\sum_{j \in M} x_{ij} = 1, \quad \forall i \in N \tag{4.50}$$

The constraint 4.50 ensures that each job is assigned to exactly one machine.

$$c_i \geq s_i + P_i, \quad \forall i \in N \tag{4.51}$$

The completion time of job i is the sum of its start time and its processing time, which is ensured by constraint 4.51.

$$s_i \geq R_i, \quad \forall i \in N \tag{4.52}$$

According to the constraint 4.52, no job starts before its release time.

$$c_i \leq D_i, \quad \forall i \in N \tag{4.53}$$

$$s_i \leq D_i - P_i, \quad \forall i \in N \tag{4.54}$$

Besides, these two constraints 4.53 and 4.54 ensure that the completion time of job i does not exceed its deadline, while job i starts early enough so that the assignment can be completed before its deadline.

$$s_i + P_i \leq s_k + M \cdot (1 - y_{ik}) + M \cdot (2 - x_{ij} - x_{kj}), \quad \forall i, k \in N, \quad j \in M \tag{4.55}$$

$$c_i \leq s_k + M \cdot (1 - y_{ik}) + M \cdot (2 - x_{ij} - x_{kj}), \quad \forall i, k \in N, \quad j \in M \tag{4.56}$$

$$s_k + P_k \leq s_i + M \cdot y_{ik} + M \cdot (2 - x_{ij} - x_{kj}), \quad \forall i, k \in N, \quad j \in M \tag{4.57}$$

$$c_i \leq s_i + M \cdot y_{ik} + M \cdot (2 - x_{ij} - x_{kj}), \quad \forall i, k \in N, \quad j \in M \tag{4.58}$$

For any jobs i and k, overlap is not allowed to occur if $i \neq k$ and both jobs are assigned to the same machine j. Constraints 4.55-4.58 constrain the sequencing of different jobs on the same machine to avoid overlap.

$$c \geq c_i, \quad \forall i \in N \tag{4.59}$$

Lastly, constraint 4.59 defines the latest completion time c as the maximum of all individual completion times.

5

Computational experiments

5.1. 3DBPP-Mixed Integer Programming

5.1.1. Context

This section utilizes the aforementioned model to conduct tests with a sample dataset. The objective is to evaluate the effectiveness of the model and derive insights that may assist in developing heuristic approaches for larger case scenarios. We have obtained a dataset from Cainiao's actual freight transportation operations, comprising various quantities of goods and Unit Load Devices (ULDs) used during booking with their contracted airline, China Southern Airlines. The test dataset is a subset of the complete data file provided by Cainiao, encompassing the packaging needs for 9 items during the last week of May 2024, from Sunday the 26th to Tuesday the 28th. Table 5.1 presents the sampled dataset used for testing, where the first column lists the delivery order numbers specifying the airfreighted goods. These goods, packed by the freight forwarder's clients into cartons, include details on weight (kg), height (cm), width (cm), and length (cm) as shown in Table 3.2. All items originated from Schiphol Airport in the Netherlands and were destined for Guangzhou Baiyun Airport in China. The dataset also specifies the release dates and deadlines for each item. To simplify computational complexity, the items are assumed to be non-fragile and can be rotated freely in any direction. We also ensured vertical stability by supporting the base at three points.

Table 5.1: A sample data set for testing model

NO	Weight (kg)	Length (cm)	Width (cm)	Height (cm)	Release date	Due date
DEPALLET2405240001	151.296	120	100	140	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
DEPALLET2405240002	114.026	120	100	140	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
DEPALLET2405240003	102.343	120	100	140	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
NLPALLET2405240001	134.318	120	100	160	2024/5/26 12:00:00 PM	2024/5/26 18:00:00 PM
NLPALLET2405240002	133.282	120	100	160	2024/5/26 12:00:00 PM	2024/5/26 18:00:00 PM
NLPALLET2405240003	184.822	120	100	160	2024/5/26 12:00:00 PM	2024/5/26 18:00:00 PM
FRPALLET2405270001	139.29	120	100	140	2024/5/27 12:00:00 PM	2024/5/28 18:00:00 PM
FRPALLET2405270002	237.07	120	100	140	2024/5/27 12:00:00 PM	2024/5/28 18:00:00 PM
FRPALLET2405270003	172.68	120	100	140	2024/5/27 12:00:00 PM	2024/5/28 18:00:00 PM

For the containers, specifically, we utilize two common types of ULDs: the AAP and AMA types, which are compatible with the lower and main decks of China Southern Airlines' Boeing 777 freighters, respectively. Table 5.2 details their main characteristics, and Figure 5.1 illustrates these containers, both of which are complete parallelepipeds.

Table 5.2: Data of AAP and AMA ULDs

Type	Length (cm)	Width (cm)	Height (cm)	Capacity (kg)	Cost (euro/kg)
AAP	317.5	223.5	162.6	4626	0.35
AMA	317.5	243.8	243.8	6800	0.35

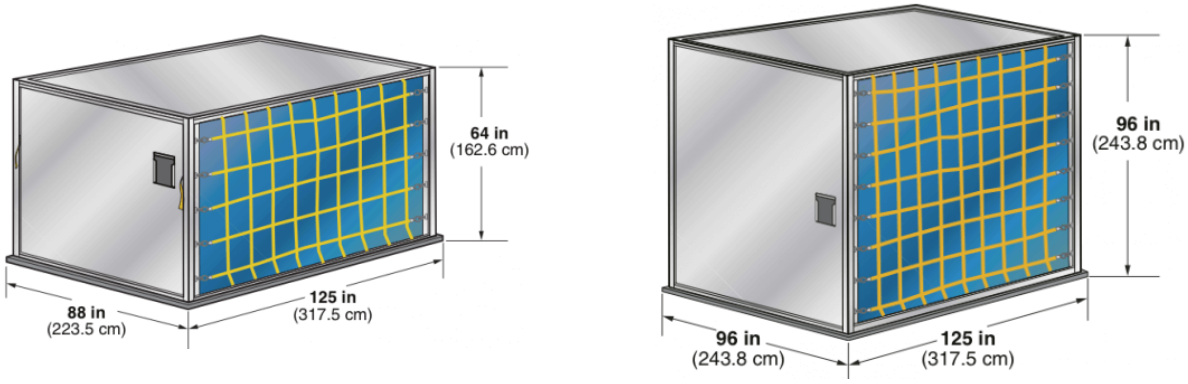


Figure 5.1: AAP ULD on the two pictures on the left-hand side and AMA ULD on the two pictures on the right-hand side [34]

Regarding time-related parameters, each item is loaded within a consistent duration of 15 minutes; Additionally, any delay in loading an item is assigned a weighted factor P that adjusts the solution based on its contribution to the overall objective function, initially set at 0.1. The maximum allowable delay for loading is 48 hours.

The computational tests were conducted on a computer with a CPU frequency of 3.20 GHz, 16.00GB of RAM, a 64-bit operating system, and equipped with Gurobi optimizer version 11.0.0 set to default parameters. To generate the results, a software program was written in Python. This program was responsible for data preparation, invoking the optimization library, and analyzing the outcomes. The outputs included the ULD costs, weighted delay terms, the start and end times of loading for each ULD, the delay times for each item (if any), and the coordinates. Additionally, three-dimensional layout diagrams were generated to visually represent the results, allowing for a clearer demonstration of the findings.

5.1.2. Computational results

Based on the data from Table 5.1 and Table 5.2, the timeline shown in Figure 5.2 outlines the time window allowed for each item and the time it is packed into the ULD, with the corresponding layout depicted in Figure 5.3. Specifically, the items shown on the timeline in Figure 5.2 (NL01, NL02, NL03, DE01, DE02, DE03, FR01, FR02, FR03) correspond to the actual boxes packed in Figure 5.3, and the AMA in Figure 5.2 refers to the type of ULD used in Figure 5.3.

From the results, it can be seen that the items are arranged to be loaded into the same ULD and that there are three delayed items (red boxes in Figure 5.3). The delay for each delayed item (NL01, NL02, NL03) is 20 hours and 15 minutes, so the cumulative delay time for the three items is 60 hours and 45 minutes. The earliest permitted start of the ULD loading is 5/27/2024 at 12:00 and lasted for 2 hours and 15 minutes. The final loading cost was €479.19.

5.1.3. Sensitivity analysis

Given that this is a multi-objective optimization problem, we apply a penalty parameter (P) as a weighting factor in the objective function to balance ULD space utilization with packing

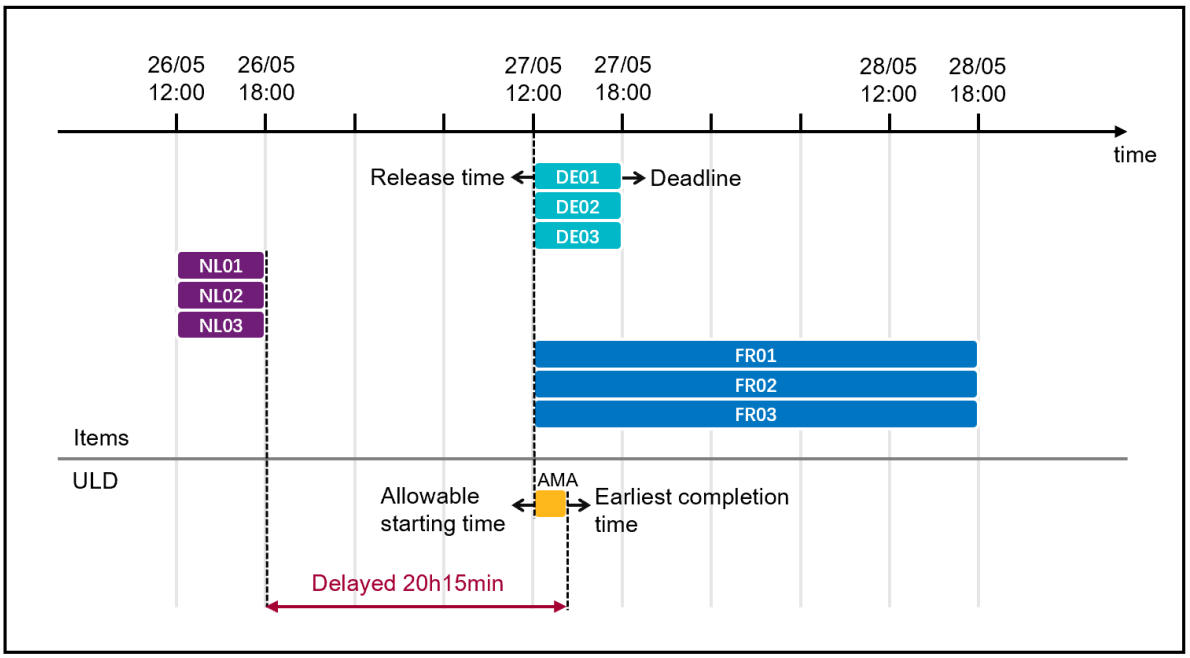


Figure 5.2: Timeline of the 9 items and a ULD

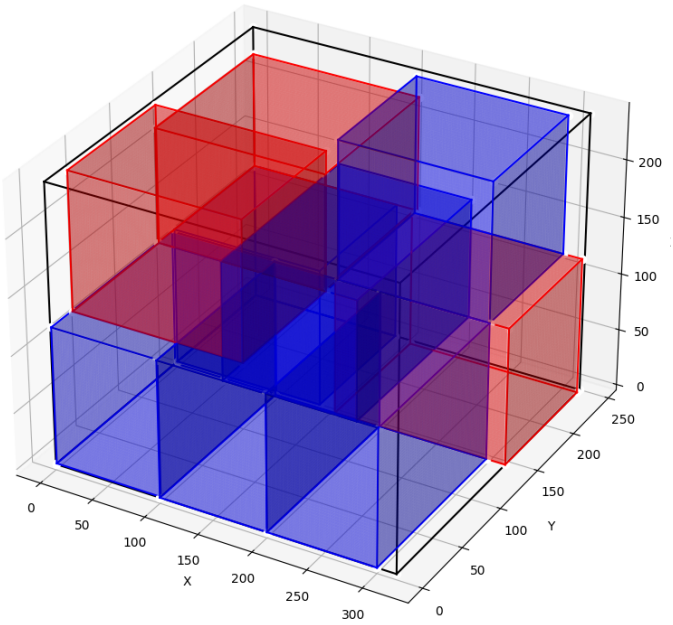


Figure 5.3: Optimal layout of the instance of 9 items in two types of ULDs

delay. The objective is to minimize the total of the unused ULD area and the delay penalty, with the penalty parameter adjusting the relative significance of each objective. By using the same dataset, the results of the effect of different penalty parameters on the number of required ULDs and the number of delayed items are obtained as in Figure 5.4. It is clear that increasing the penalty value influences the packing design, with the penalty acting as a regulator of the trade-off between minimizing the number of ULDs and minimizing delays in packing.

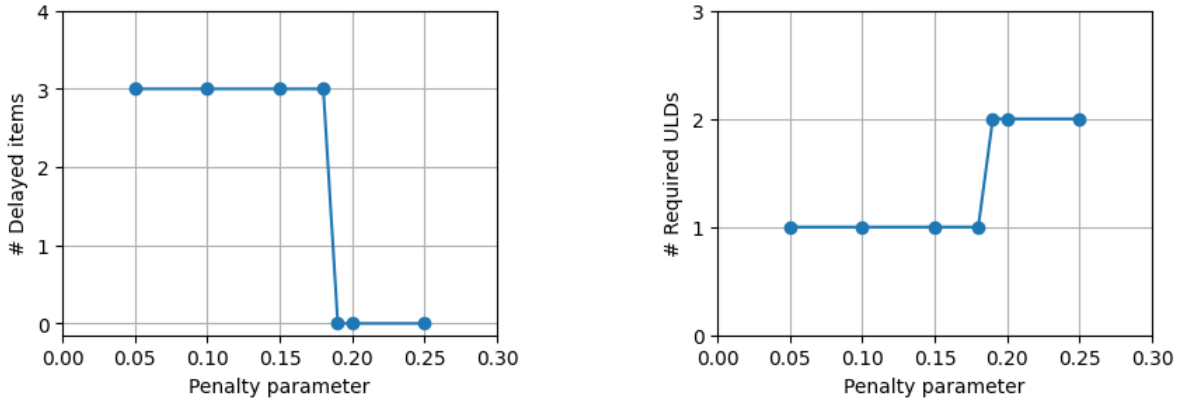


Figure 5.4: Impact of penalty parameters on the number of delayed items and ULDs required

As the penalty weight increases, a significant change in the solution is observed. For lower penalty values (0.1-0.18), the model prefers to minimize the number of ULDs utilized to only one, yet this comes at the expense of delaying three items. This finding indicates that when the penalty parameter is less than 0.19, the penalty for delaying packages is insufficient to support the deployment of extra ULDs. Once the penalty weight reaches 0.19, a clear change occurs: the model chooses to use additional ULDs (two in total), thus eliminating the delay completely. When the penalty value hits 0.19 or higher, the cost of delays outweighs the cost of using more ULDs, leading to a packing configuration that avoids delay altogether. This demonstrates that the penalty value effectively regulates the trade-off between these two conflicting objectives.

5.1.4. Model validation

In order to assess the quality of the solution, multiple experiments were conducted. The principal findings are summarized in Table 5.3. The first column presents the number of items to be loaded, the second column shows the CPU time (in seconds), the third column details the number of required/available ULDs, and the final column shows the gap, which is the relative difference between the objective value of the best feasible solution and the best known lower bound.

Different number of items & ULDs

Firstly, the number of items or the number of available ULDs was varied. Under a time constraint of 7200 seconds, results indicate that computational time increases whether more items are involved or more ULDs are available. When the number of items loaded was 9, the computation time was significantly different for different conditions of the number of ULDs available. For example, when 2 ULDs are available, the computation time is 22 seconds, while when 4 ULDs are available, the computation time increases significantly to 230 seconds. This indicates that as the number of available ULDs increases, the computational complexity rises accordingly. When the number of items increases to 10, the computation in all cases reaches the set upper time limit and no better solution can be found within the given time. It is possible that the increase in the number of items, which involves the use of an extra ULD, leads to a

Table 5.3: Validation results for MIP

# Items	Time(s)	# Required/Available ULDs	GAP	Type of constraints
9	22	1/2	0%	27 quadratic constraints
9	15	1/3	0%	36 quadratic constraints
9	230	1/4	0%	45 quadratic constraints
9	3	1/2	0%	0 quadratic constraint
9	40	1/3	0%	0 quadratic constraint
9	25	1/4	0%	0 quadratic constraint
10	7200	1/2	37.90%	30 quadratic constraints
10	7200	1/2	37.90%	0 quadratic constraints
10	7200	1/4	50%	0 quadratic constraints

larger problem size and a significant increase in the difficulty of solving the model.

Quadratic & linear constraints

The experiment rewrote all the quadratic constraints in the model as linear constraints. Taking constraint 4.47 as an example, the product of the variables end_j and n_j leads to multiple quadratic constraints in this model, which affects the computation time. To eliminate the quadratic terms from its formulation, the auxiliary variable y_{ij} is used, and the constraints are linearized. The following new linear constraints are introduced:

$$y_{ij} \leq K \cdot n_{ij}, \quad \forall i \in N, j \in M \tag{5.1}$$

$$y_{ij} \leq end_j, \quad \forall i \in N, j \in M \tag{5.2}$$

$$y_{ij} \geq end_j - (1 - n_{ij}) \cdot K, \quad \forall i \in N, j \in M \tag{5.3}$$

$$y_{ij} \geq 0, \quad \forall i \in N, j \in M \tag{5.4}$$

where K is a constant large enough to ensure the validity of the above constraint. The final linearized constraint after replacement is shown in constraint 5.5. Experiments using the above-replaced constraints show that the computational time of the model tends to be shorter in the absence of quadratic constraints, which verifies that the linearized constraints can significantly reduce the computational complexity. On the other hand, the models with quadratic constraints have higher GAP values when the time limit is reached. This indicates that the quadratic constraints increase the complexity of the problem and thus affect the performance of the model.

$$d_i \geq \sum_{j \in M} y_{ij} - DDL_i, \quad \forall i \in N \tag{5.5}$$

The experimental results show that for smaller problems (e.g., 9 items), this mathematical model of 3DBPP can find the optimal solution in a reasonable amount of time (GAP of 0%), but when the problem size is slightly increased (e.g., 10 items), the quality of both the solution time and the solution quality decreases significantly. The solver fails to discover the best

solution within the time constraint of 2 hours, particularly when there are more abundant ULD resources, such as 4 available ULDs (GAP up to 50%).

This circumstance demonstrates that existing exact algorithms confront a significant rise in computing time as the problem size grows, and even when linearization techniques are applied to minimize computational complexity, it remains difficult to discover an optimal solution in an acceptable amount of time. This constraint is especially evident when dealing with large-scale instances in real-world applications, where rapid delivery of high-quality solutions is frequently necessary. As a result, new algorithmic models that use heuristics to obtain a near-optimal solution suited to large-scale problems may be explored.

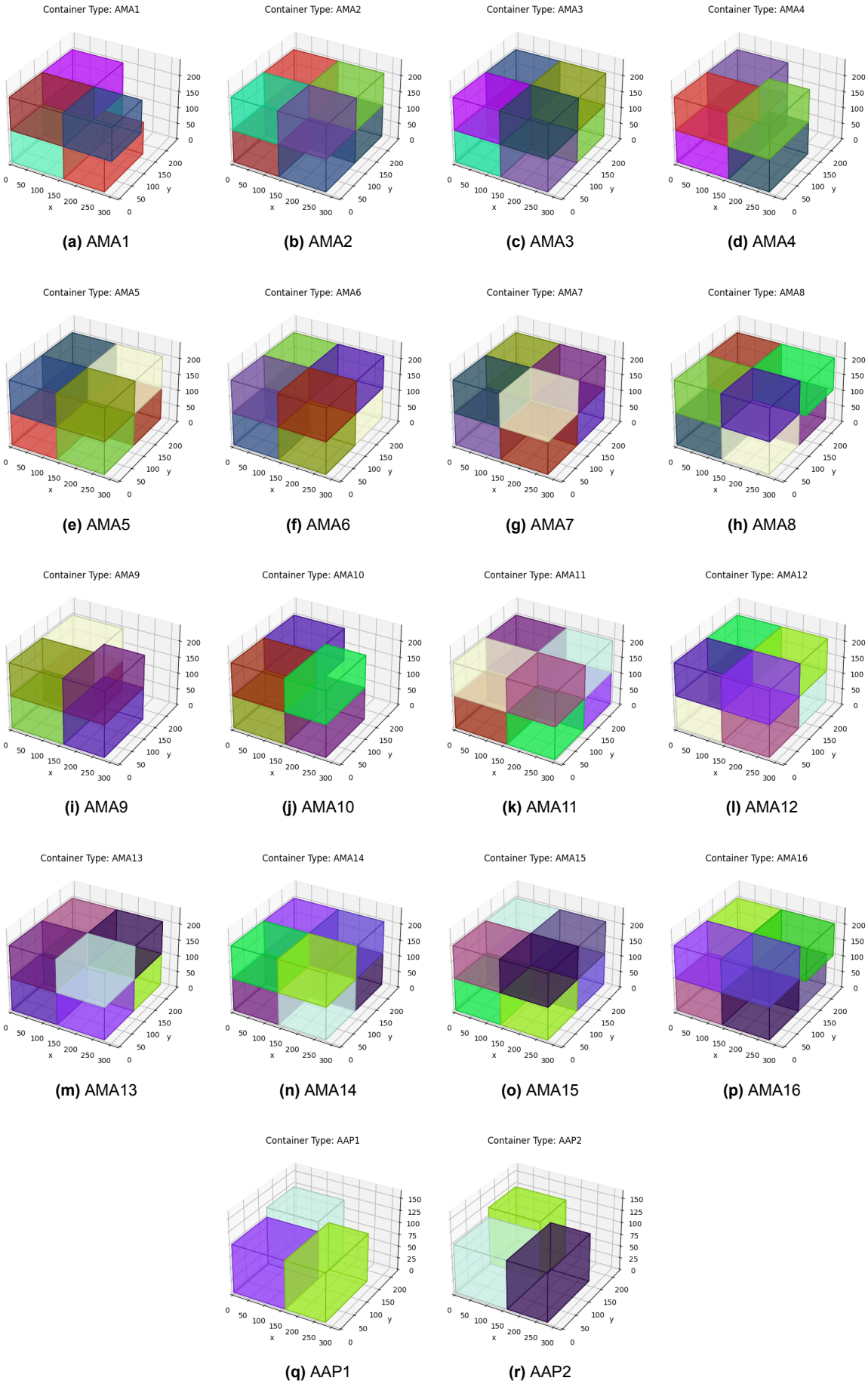
5.2. 3DBPP-Extreme Point Heuristic Algorithm

5.2.1. Context

This section uses the heuristics algorithm described above to test a larger sample dataset. The item and ULD parameters remain unchanged from those used in the previous MIP model, but the number of items increases from 9 to 126 in terms of packaging requirements, as detailed in Appendix A. In addition, because of the added assumption that packing can be done in parallel no matter how many ULDs there are. If there is an item delay, then the delay is equal to the difference between the due date of the item and the theoretical earliest completion time of the ULD.

5.2.2. Computational results

The implementation of EP heuristic algorithm to solve large-scale 3DBPP using Python and the layout results are shown in Figure 5.5. 126 items are scheduled to be loaded into 18 ULDs and no items are delayed in loading. Appendix B lists for each ULD the items packed, the duration of the load, the latest release time of the loaded items, the earliest completion time, the earliest due date (i.e., the earliest deadline of the packed items), and the cost spent per ULD.



(a) AMA1 **(b)** AMA2 **(c)** AMA3 **(d)** AMA4

(e) AMA5 **(f)** AMA6 **(g)** AMA7 **(h)** AMA8

(i) AMA9 **(j)** AMA10 **(k)** AMA11 **(l)** AMA12

(m) AMA13 **(n)** AMA14 **(o)** AMA15 **(p)** AMA16

(q) AAP1 **(r)** AAP2

Figure 5.5: Solution of the instance of 126 items in two types of ULDs

5.2.3. Sensitivity analysis

Overhang threshold

The presence of overhang in item loading can be seen from the AMA1 (a-plot) in Figure 5.5. As stated in subsection 4.2.2, the optimal percentage of overhang allowed needs to be recognized in order for item placement to be stably supported. To determine the optimal allowable overhang percentage that balances packing efficiency with stability, we conducted a series of experiments varying the overhang threshold from 10% to 50% in increments of 10%. The number of Unit Load Devices (ULDs) required at each threshold was recorded, as summarized in the accompanying Figure 5.6.

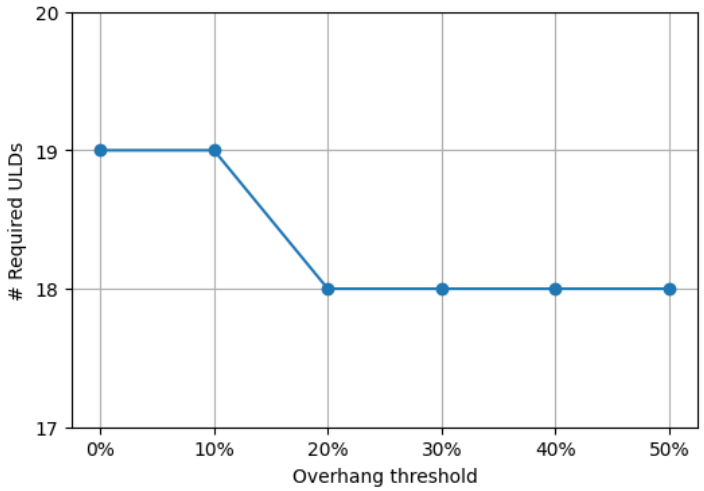


Figure 5.6: Results of experiments for overhang threshold

The results show that at the lower threshold of 0% to 10%, the number of required ULDs remained consistent at 19. This indicates that, within this range, the permitted overhang has little effect on total loading efficiency. However, by increasing the threshold to 20%, the number of required ULDs decreased to 18, demonstrating a significant improvement in packing efficiency as more overhang was permitted. Increasing the threshold to 30% resulted in no further decreases in the number of ULDs required, suggesting that packing efficiency had plateaued. These findings suggest that the optimal overhang threshold likely lies between 10% and 20%. This range increases packing efficiency by lowering the amount of ULDs required while maintaining acceptable stability. Beyond this point, further increasing the overhang percentage does not yield additional benefits and may risk compromising the structural stability of the load.

Item size and ULD types

Volume Utilization is a key measure of the efficiency of the loading algorithm, reflecting the extent to which the available space inside each ULD is effectively used. Volume utilization verifies loading effectiveness by calculating the ratio of the total volume of items to the total usable volume of the ULD. It can provides important insights into the interaction between cargo dimensions and ULD types. In this study, four distinct item size categories were analyzed, including two commonly used Cainiao packaging configurations, as well as a smaller size for compact items and a larger size for oversized goods, as shown in Table 5.4. Besides, Five ULD types differ in length, width, height, and maximum load capacity, as shown in Table 5.5. These differences significantly affect the utilization of loading space. By evaluating the impact of these variations on the efficiency of space utilization, this analysis explores the adaptability of different ULD types to diverse cargo scenarios.

Table 5.4: Data of 4 different types of items

Item type	Weight (kg)	Length (cm)	Width (cm)	Height (cm)
Compact	110	100	100	100
Standard-Small	140	120	100	140
Standard-Large	150	120	100	160
Oversized	160	140	120	160

Table 5.5: Data of 5 different types of ULD [34]

Type	Length (cm)	Width (cm)	Height (cm)	Capacity(kg)
ALP	317.5	153.4	162.6	3176
AAP	317.5	223.5	162.6	4626
AMP	317.5	243.8	162.6	5103
AMA	317.5	243.8	243.8	6800
AGA	605.8	243.8	243.8	11340

The experimental results are shown in Figure 5.7, which demonstrate that small and medium-sized items collectively exhibit relatively high volume utilization due to their greater flexibility in arrangement, which minimizes void space during packing. For smaller items, the compact dimensions allow efficient placement across all ULD types, with larger ULDs such as AMA and AGA achieving the highest utilization rates. Similarly, medium-sized items maintain good adaptability but begin to encounter constraints in narrower ULDs like ALP and AAP, where gaps may emerge due to dimensional mismatches. However, medium-sized items perform well in wider and taller ULDs, particularly AMA and AGA, which mitigate these constraints and enable better packing configurations. The combined results for smaller and medium-sized items illustrate that volume utilization is highly sensitive to both ULD dimensions and the size category of the cargo, with larger ULDs generally providing more consistent efficiency.

Oversized items introduce further complications in terms of sensitivity to ULD selection. Due to their substantial dimensions, they occupy a significant proportion of any ULD's internal space, which can limit opportunities for efficient arrangement. Interestingly, the results suggest that smaller ULDs, such as ALP and AAP, sometimes achieve higher volume utilization for oversized items compared to larger ULDs. This outcome reflects the proportional alignment between the dimensions of oversized items and the smaller ULDs, which reduces void space and improves utilization. In contrast, larger ULDs like AGA demonstrate lower utilization rates for oversized items, as the excess internal space exacerbates underutilization.

Among the five ULD types, AMA demonstrates the most balanced performance across all item size categories. Its moderate dimensions, which are wider and taller than those of ALP and AAP but less expansive than AGA, allow for consistent adaptability to varying item sizes. This versatility is particularly evident in its ability to achieve high utilization for both small and medium-sized items while maintaining reasonable efficiency for larger items. The sensitivity of AMA's utilization rates to item size is relatively low compared to other ULDs, making it the most practical choice for mixed cargo shipments that require flexibility and efficiency. By contrast, AGA, while capable of accommodating large quantities of cargo, is more sensitive to item size variations, with efficiency declining significantly for oversized items.

5.2.4. Model validation

The experiments in this section aim to validate the model and test its performance on different scenarios or datasets. This section verifies the results and proves their validity through several

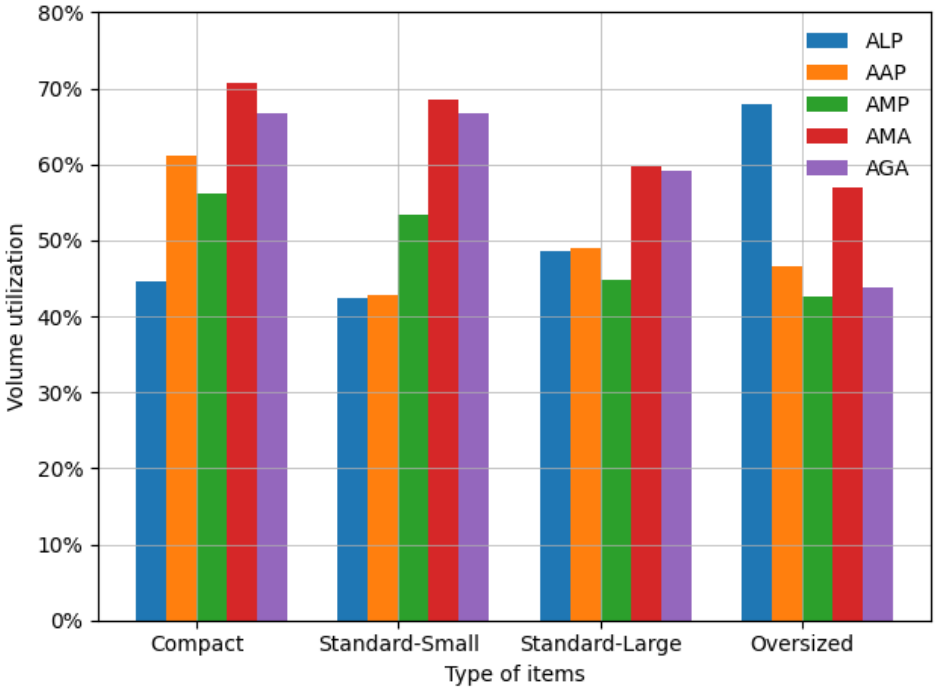


Figure 5.7: Results of volume utilization for item size and ULD types

methods:

Comparison of EPH and MIP

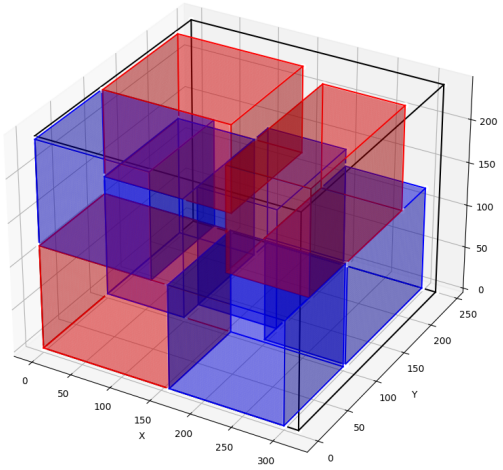
Mixed Integer Programming and Extreme Point heuristic algorithms have been used to solve the 3DBPP above, and their performance is comparatively analyzed in this section. Despite the differences in their modeling constraints, the results remain comparable due to their shared optimization objective.

In particular, the MIP model ensures stacking stability through a three-point support rule, while the EPH method explicitly incorporates a constraint on the minimum overhang ratio of items. This requires each item to be supported by a specific proportion of surfaces on either the ULD floor or other items beneath it. This distinction reflects a variation in how the two algorithms define solution feasibility during the optimization process. The MIP model's three-point support rule is relatively flexible, allowing for broader exploration of the solution space and increasing the likelihood of identifying layouts with higher space utilization. In contrast, the EPH method's stricter support constraints limit the solution space, which in turn makes it more challenging for the algorithm to find optimal solutions under certain conditions.

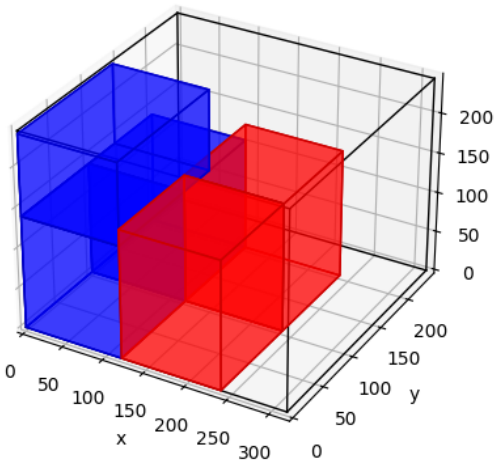
To ensure a fair comparison between the two algorithms, this study introduces appropriate relaxations to the constraints of the EPH model when evaluating its performance. For instance, the pre-sorting of item loading order was removed, and the allowable overhang area was relaxed to 40%. These adjustments were designed to align the EPH solution environment more closely with the assumptions of the MIP model, thereby reducing the impact of differing constraint definitions on the comparison results.

Under these adjustments, in the context of a small dataset, if the items are allowed to rotate arbitrarily, as shown in (a) of Figure 5.8, the MIP model uses only one ULD, and the layout of the items presents a more compact and balanced distribution, which is able to utilize the space inside the ULD more efficiently. In contrast, the EPH algorithm makes it difficult to find

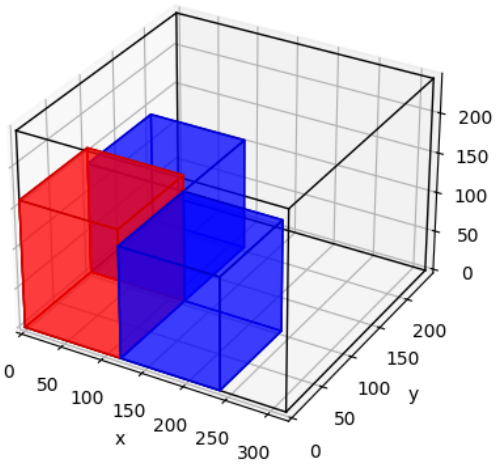
an optimal solution and needs to use two ULDs when loading the same items. The loading results (b and c) show that most of the space is wasted. Even after relaxing the conditions, EPH still struggles to achieve a space utilization efficiency comparable to MIP.



(a) AMA1 loading result obtained with MIP



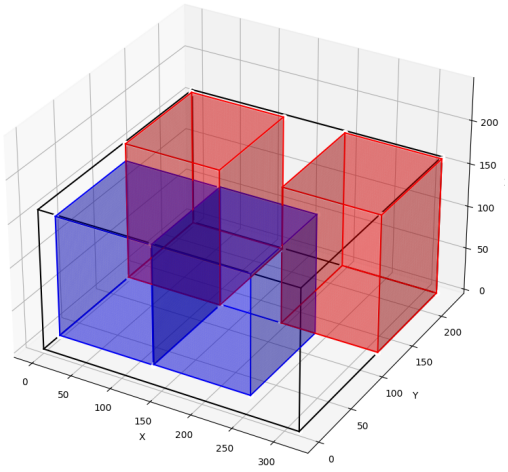
(b) AMA1 loading result obtained with EPH



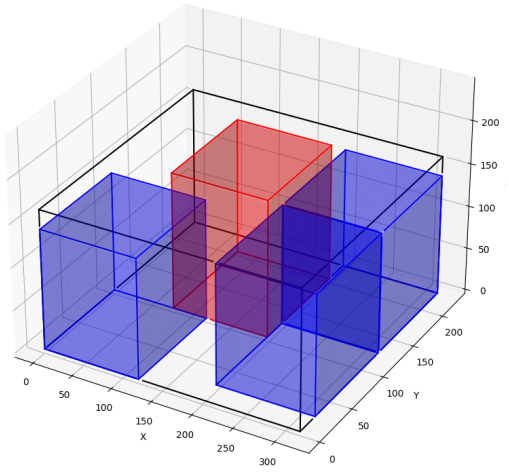
(c) AMA2 loading result obtained with EPH

Figure 5.8: Results of the MIP method (top row) and the EPH method (bottom row) with items that can be freely rotated

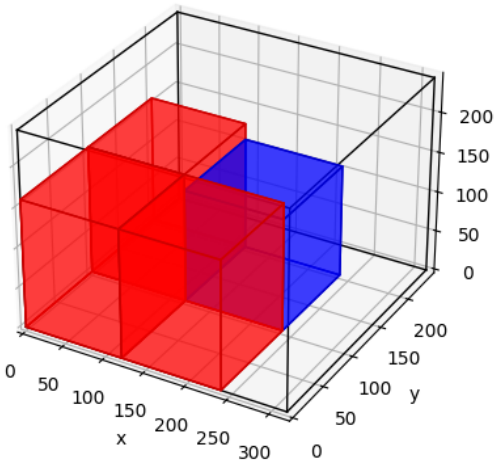
However, the EPH algorithm and the MIP model perform similarly in terms of the number ULD when items must be placed top-side up. Figure 5.9 illustrates the loading results of these two methods for the same instance. The EPH method uses a larger AMA-type ULD, which is more compact but results in more wasted space. The MIP method, on the other hand, uses smaller AAP-type ULDs, making it more space-efficient, but the layout is relatively loose and slightly unstable. In addition, both methods also perform consistently in dealing with delayed packing decisions by delaying the packing of three items (boxes filled in red) to be loaded with other items, thus saving the use of one ULD and improving space utilization. This result also aligns with the practical situation in air transport where certain goods need to be placed in a specific orientation.



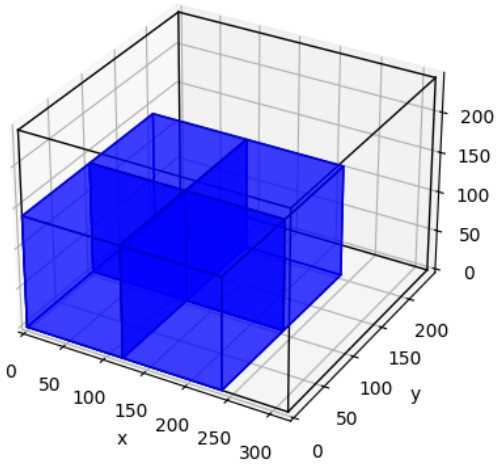
(a) AAP1 loading result obtained with MIP



(b) AAP2 loading result obtained with MIP



(c) AMA1 loading result obtained with EPH



(d) AMA2 loading result obtained with EPH

Figure 5.9: Results of MIP Method (top row) and EPH Method (bottom row) in the context of items that must be top-side up

Running time analysis

As the size of the dataset increases, the difference in performance between the two methods becomes apparent. The MIP model completely loses its ability to solve efficiently when dealing with large-scale datasets due to the dramatic increase in computational complexity, and more powerful computing equipment is required to obtain the results. In contrast, the EPH algorithm performs outstandingly well and is able to quickly obtain a solution in a shorter period, which is particularly important for efficient computation in practical applications. In the case of Cainiao, for example, which carries a large amount of cargo by air every week, fast computation results are essential to meet the needs of its efficient operations.

To evaluate the proposed EPH algorithm, we employed a large number of problem instances with standard-small-size item counts ranging from 5 to 5000, and the experimental results are shown in Table 5.6. The figure illustrating the relationship between the number of items and the running time (Figure 5.10) shows that the running time tends to grow linearly with the number of things. This indicates that the algorithm's complexity is linearly scalable when deal-

ing with larger-scale problems, which is a desirable performance for the large-scale 3DBPP, demonstrating that the algorithm can deal with a large number of items without increasing computation time exponentially.

Table 5.6: Validation results for EPH

# Items	Time (s)	# ULDs	# Lb ULDs	Deviation
5	0.4	1	1	0%
15	0.4	2	2	0%
25	1.2	4	3	25%
30	1.3	4	3	25%
50	2.6	7	5	29%
100	4.9	13	9	31%
150	7.7	19	14	26%
200	10.6	25	18	28%
300	15.9	38	27	29%
500	26.0	63	45	29%
1000	53.0	125	89	29%
2500	143.6	313	223	29%
5000	321.5	625	446	29%

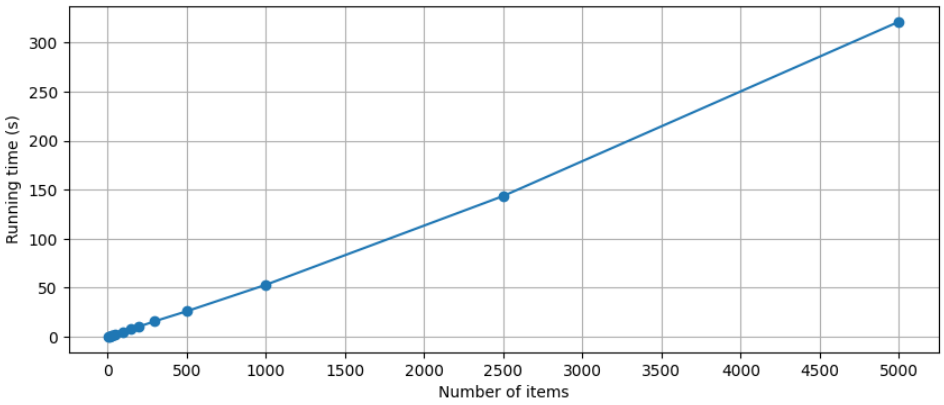


Figure 5.10: Result of running time versus number of items

ULD quantity deviation analysis

In the ULD quantity deviation analysis, we utilize the deviation formula (Equation 5.6) to measure the difference between the actual number of ULDs (Unit Load Devices) used and the theoretical lower bound for assessing the effectiveness of the loading algorithm [26]. Specifically, it indicates the percentage by which the actual number of ULDs used exceeds the theoretical minimum number of ULDs. A lower percentage of deviation calculated indicates that the algorithm is approaching the theoretical optimal solution, while a higher deviation indicates a larger distance between the actual result and the theoretical optimal solution.

$$Deviation = \frac{b - lb}{b} \times 100\%, \tag{5.6}$$

where b is the actual number of ULDs used, lb is the lower bound of the theoretically calculated number of ULDs, and lb is defined by Equation 5.7-5.9, which combines the estimation of the lower bound based on weight and volume, respectively.

$$lb_1 = \frac{\sum_{i \in N} weight_i}{MaxW}, \tag{5.7}$$

where lb_1 represents the minimum number of ULDs required in terms of the total weight of the item, and $MaxW$ is the maximum weight allowed of all ULDs.

$$lb_2 = \frac{\sum_{i \in N} v_i}{MaxV}, \tag{5.8}$$

where lb_2 indicates the minimum number of ULDs required based on the estimated total volume of the item, and $MaxV$ is the maximum volume of all ULDs.

$$lb = max(lb_1, lb_2) \tag{5.9}$$

The formula 5.9 implies that the greater of the two values, weight and volume, needs to be taken as the theoretical lower bound (lb). This is because, while the item's volume can fit into a single ULD, it may exceed the ULD's load capacity, and vice versa.

The Figure 5.11 illustrates the relationship between the number of actual ULDs utilized and the number of lower bounds obtained using Equation 5.6-Equation 5.9. In practice, a low deviation is critical for logistics or warehousing scenarios since it indicates that the algorithm is effectively exploiting ULDs in real-world applications. Table 5.6 displays the values for the deviations of the two quantities. It is clear that the number of actual ULDs is always more than the theoretical lower boundary, and the deviation progressively stabilizes as the number of items grows. When the number of items exceeds 25, the gap is between 25% and 31%. As the number of items rises, the deviation ultimately stabilizes at around 29%. This is because the lower limit is predicted based on the weight and volume of the items, without accounting for many factors. However, the shape, arrangement, and space allocation of the items in the actual loading process may require more ULDs than the theoretical lower limit. The fact that its deviation remains generally consistent over large sample sizes demonstrates that it can produce steady and acceptable results in real-world applications.

Volume utilization validation

To validate the loading efficiency of the model, standard-small-size items with length, width, and height of 120cm, 100cm, and 140cm were selected to be put into five different types of ULDs, including ALP, AAP, AMP, AMA, and AGA. Then the space utilization for different quantities of cargo was tested.

The results of the experiments are shown in Figure 5.12, where the space utilization of each ULD type gradually stabilizes as the number of loads increases. Among them, the AMA and AGA types show the most outstanding performance, with the space utilization reaching and maintaining at about 71% when the number of items increases to 200 and above. This efficient space utilization is mainly due to their larger width and height as well as higher maximum loading capacity. The results show that the heuristic algorithm proposed in this study has good stability when loading common-sized cargoes of these ULD types, and can maintain an efficient loading effect. However, from the perspective of space utilization, this result also reveals some room for improvement.

This might be due to the fact that the heuristic approach described in this paper fails to generate optimal outcomes. The heuristic method will first arrange the items in the default placement

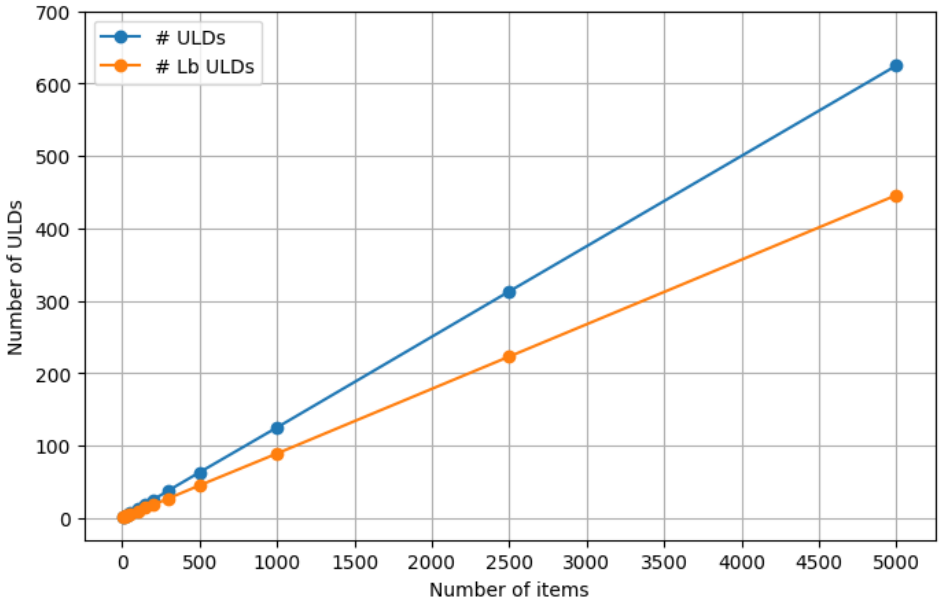


Figure 5.11: Difference between the actual number of ULDs used and the lower bound

direction, and only if the items cannot fit in the default direction will it consider rotating the items to make modifications. Although this placement strategy simplifies the implementation of the algorithm, one major problem is that the items are placed independently of each other. It means that after the location of the front item is determined, subsequent items' placement will no longer allow the front item to be modified, resulting in fewer placement possibilities for the following items. When irregular gaps make it difficult to accommodate subsequent items, more ULDs need to be used to load the subsequent items. This directly leads to an increase in the number of ULDs used and reduces the overall space utilization.

The test results for different ULD types also validate this inference. In contrast, larger-sized ULDs such as AMA and AGA were able to accommodate more cargo and therefore had relatively less adverse effects on their placement strategies, ultimately achieving higher space utilization. Whereas for smaller ULDs such as ALP and AAP, their space utilization was relatively low, remaining at around 42% to 43% on average. This result may be related to the smaller width dimensions of these two types of ULDs, limiting the efficient utilization of their loading space. This difference suggests that the size and capacity of ULDs play a key role in loading efficiency, with larger ULDs not only increasing cargo holding capacity but also mitigating to some extent the space wastage associated with placement strategies.

5.3. Build-up Scheduling Problem

5.3.1. Context

The inputs for the BSP model originate from the previous section, in which we developed a bin-packing solution using the EPH. The solution specifies the jobs that need to be handled, i.e., the ULDs to be loaded during the two-day scheduling period. Each job includes specific requirements such as release time, processing time, and deadline, as shown in Appendix B.In addition, it is expected that there are three parallel machines (loading stations). The BSP model then develops an optimum machine scheduling plan based on these inputs, ensuring that no jobs overlap on the same machine and that all constraints such as job release times and deadlines are satisfied. The goal is to assign and schedule 18 ULDs to three loading

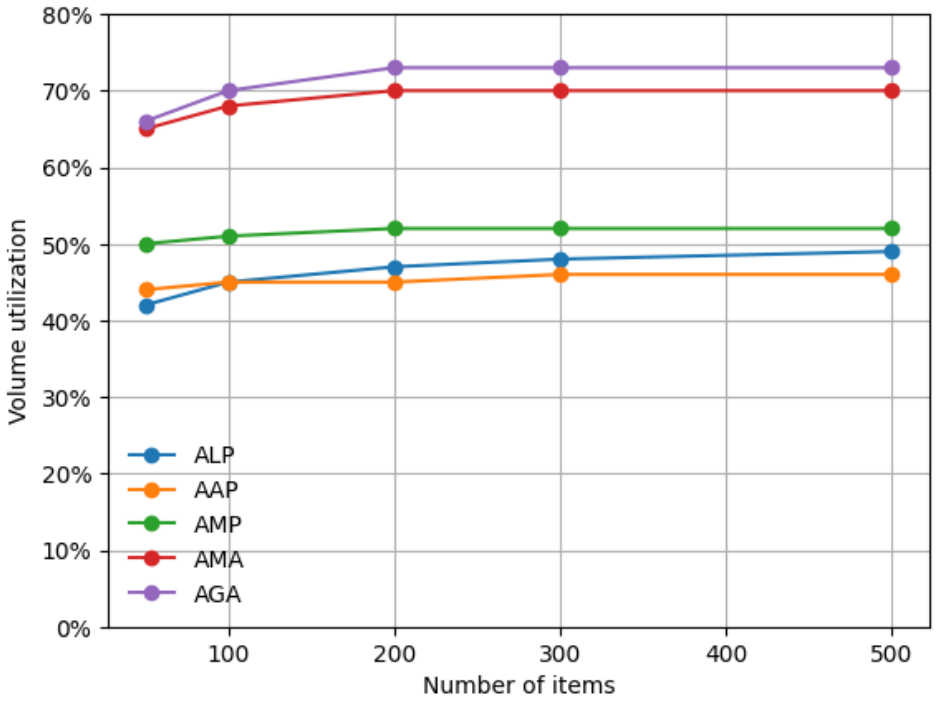


Figure 5.12: Results of volume utilization

stations over two days while minimizing the overall completion time for all loading stations.

5.3.2. Computational results

In this section, we analyze the computational results of the Build-up Scheduling Problem (BSP) model over a two-day scheduling period. The results of the computational experiment are shown in the Figure 5.13 for two consecutive days, May 27, 2024, and May 28, 2024.

From the results, the total job duration for day 1, 27 May 2024, is 6 hours and the ULDs are distributed across all three workstations with no overlap or violation of job deadlines. Day 2 had a job interval of 4 hours and 45 minutes, and the ULDs were distributed efficiently among the loading stations. The final total completion time was 28 May 2024 at 16:45.

It is worth stating that AMA10 was released on 27 May 2024 at 12:00 PM. Its deadline was set for 28 May 2024 at 18:00. However, in the above result, AMA10 was processed on 27 May 2024, well before its deadline. This decision to process earlier may be attributed to machine load balancing optimizations. Because of the need to meet deadlines, avoid task overlap, and minimize overall completion time, the model distributes ULDs across workstations in a balanced manner to optimize performance and avoid bottlenecks. It can be seen that scheduling AMA10 earlier results in a smoother run and reduces conflicts with other ULDs scheduled to be processed on 28 May, such as AMA11 and AMA12. Then all other ULDs are processed on the day of their respective deadlines. For example, ULDs like AMA11 through AMA16 are all released and scheduled for 28 May 2024, and their respective processing times are all completed by 18:00 on the day of the deadline. This strategy of processing ULDs with flexible deadlines (such as AMA10) ahead of time while aligning other ULDs with their strict deadlines helps to achieve the goal of minimizing delays and ensuring that no ULD exceeds its due date.

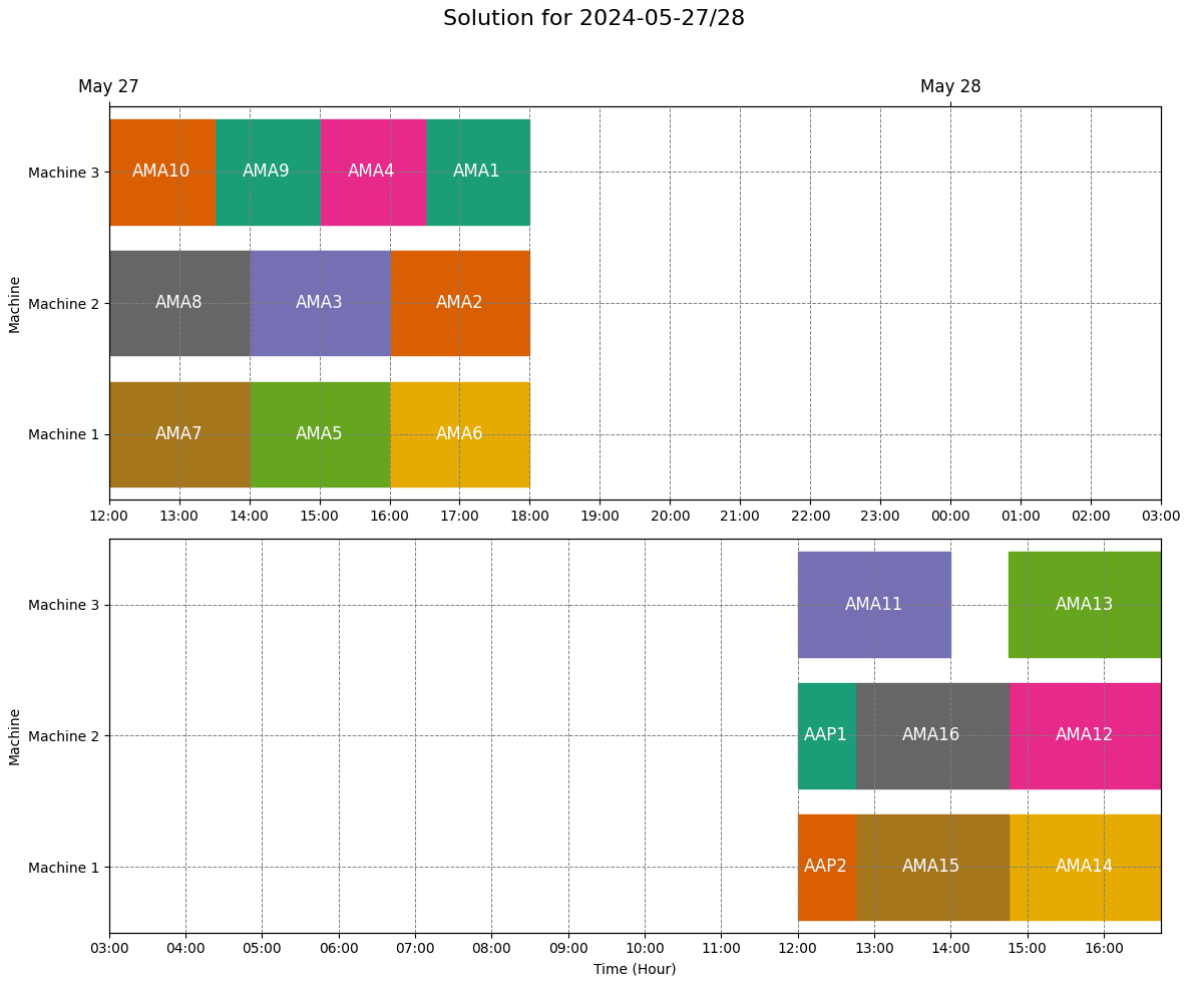


Figure 5.13: Solution of the instance of 18 ULDs in 3 loading stations

5.3.3. Sensitivity analysis

The sensitivity analysis conducted on the number of available workstations for the scheduling of 18 jobs is shown in Table 5.7, which provides critical insights into the relationship between resource allocation and operational efficiency.

With only two workstations, the problem becomes infeasible, emphasizing the necessity of maintaining a minimum resource threshold to meet scheduling constraints such as processing times, release times, and due dates. As the number of workstations increases, the makespan steadily decreases, reflecting improved efficiency; however, the rate of improvement diminishes significantly beyond five workstations. From seven workstations onward, the makespan stabilizes at 14:00. As the number of workstations increases, the time steadily decreases, reflecting increased efficiency: when the number of workstations is increased from 6 to 7, the time improves to 14:00. While the analysis demonstrates that further increases in the number of workstations (e.g., allocating 8 workstations) do not provide additional benefits. This results in underutilized resources, which could be better allocated elsewhere or minimized to reduce costs.

Table 5.7: Results of experiments for number of work stations

# Work stations	Makespan
2	infeasible
3	2024/5/28 16:45:00 PM
4	2024/5/28 16:00:00 PM
5	2024/5/28 16:00:00 PM
6	2024/5/28 15:30:00 PM
7	2024/5/28 14:00:00 PM
8	2024/5/28 14:00:00 PM

6

Conclusion

6.1. Discussion

The proposed framework, integrating the Air Cargo Palletization Problem and the Build-up Scheduling Problem, offers a comprehensive solution to address key logistical challenges in air cargo operations. By connecting packing and scheduling decisions into a unified pipeline, the framework provides several managerial insights and demonstrates significant potential for real-world implementation.

From a managerial perspective, this framework highlights the interplay between strategic planning and operational execution. In practical air cargo operations, decisions made during the packing stage (e.g., ULD selection, cargo arrangement) have downstream implications for scheduling and resource allocation. The sequential nature of the framework ensures that these dependencies are accounted for, creating a structured decision-making process. The sequential approach simplifies implementation in real-world settings, particularly in logistics systems where decision-making is distributed across departments or teams. For example, in an air cargo hub, the packing and scheduling processes are often managed separately, with limited real-time communication. The proposed framework introduces a structured methodology to integrate these stages without requiring significant changes to existing workflows. By implementing this design, logistics managers can improve coordination between teams, reduce bottlenecks, and enhance overall system efficiency.

A more detailed discussion of each phase is provided in the following sections.

6.1.1. Air Cargo Palletization Problem

In cross-border e-commerce logistics, the diversity of goods and differences in delivery timelines create challenges for improving loading efficiency and managing delivery scheduling. In air cargo management, enhancing loading efficiency directly impacts transportation costs and operational resource utilization. This study validates the performance of different ULD types for different cargo scenarios through a series of three-dimensional packing experiments, providing significant management insights. The findings reveal that larger ULDs (e.g., AMA and AGA) are more effective for handling shipments that involve small-sized or standard-sized goods, as their greater dimensions allow for better space utilization and adaptability to diverse cargo configurations. However, for oversized items, smaller ULDs such as ALP can achieve surprisingly better utilization due to proportional dimensional alignment. In real-world logistics operations, shipments often consist of a mix of item sizes rather than uniform cargo, making

larger ULDs such as AMA a versatile choice for achieving balanced space utilization and operational efficiency. From a practical management perspective, these insights can inform the strategic selection of ULDs. Air cargo companies can benefit from prioritizing larger ULDs for mixed-size shipments to maximize loading efficiency while deploying smaller ULDs strategically for oversized cargo to address specific requirements. A flexible ULD allocation strategy that reflects the predominance of mixed-size goods in logistics operations can optimize resource allocation, reduce costs, and enhance service reliability.

A deeper analysis of the experimental results reveals differences between MIP and EPH algorithms under varying shipment volumes. While MIP can deliver optimal solutions for small datasets, its computational efficiency drops significantly for large datasets, making it unsuitable for large-scale transport scenarios. In contrast, the EPH algorithm excels in terms of computational efficiency. While heuristic algorithms in the traditional literature (e.g., the study by Paquay, Schyns, and Limbourg (2016)) have an efficiency advantage in terms of computational time, the validation in this paper also shows the tractability of the EPH algorithm in large-scale applications. Heuristic algorithms like EPH may not always achieve the optimal solution but demonstrate better time efficiency and computational stability for large-scale packing problems. This suggests that air cargo companies can select algorithms based on business needs: MIP's precise solutions can enhance loading efficiency for small-scale shipments, while EPH's faster computations make it more suitable for large-scale operations. This flexibility in algorithm selection supports more targeted decision-making for different scales of cargo tasks.

Additionally, in cross-border e-commerce logistics, Cainiao's shipments often involve diverse sizes and shapes, along with varying delivery deadlines. Traditional 3D bin packing models have limitations when addressing such requirements, especially since they lack consideration of delivery timing. While previous research, such as that by Chen, Lee, and Shen (1995), Paquay, Schyns, and Limbourg (2016), and Paquay, Limbourg, and Schyns (2018), focused on improving space utilization, it often neglects the crucial aspect of delivery time. In a global supply chain, delivery timeliness and accuracy directly influence customer satisfaction and brand reputation. To address this gap, this study incorporates item availability and delivery deadlines into the 3D bin packing model. This enhancement improves both space utilization and logistics efficiency, making the model more practical and effective.

The introduction of time constraints highlights the importance of multi-objective optimization in logistics management. By using a delay penalty parameter, the model can strike a balance between maximizing ULD space usage and meeting time constraints. This provides a flexible decision-making tool for logistics managers. For example, during peak cargo periods, increasing the delay penalty can ensure timely delivery of critical goods, optimizing overall resource allocation and reducing potential financial losses from delays. In scenarios with high cargo volumes with less strict timing requirements, reducing the penalty can minimize ULD use and lower transportation costs. This adjustment based on the delay penalty parameter offers flexible strategies for cargo scheduling, enabling managers to make decisions that align with business needs and optimize resource utilization.

6.1.2. Build-up Scheduling Problem

An optimization model based on the Parallel Machine Scheduling Problem is used in this paper to address the loading scheduling requirements from Cainiao. By scheduling jobs across multiple parallel workstations, the model successfully optimises the order and timing of ULD assembly, avoiding delays in the loading process and improving the overall efficiency. The results show that the BSP model can ensure that all workstations are load-balanced and that

the assembly process of each ULD fits into its time window, minimizing the waste of resources during the loading process.

From the managerial perspective, the implementation of the BSP model provides important insights into the scheduling management of air cargo. First, the flexibility of the BSP model is reflected in the adaptation of different job priorities. For example, the advanced processing of AMA10 shows that the model can reasonably arrange ULDs with flexible delivery times while balancing the job loads, making the overall scheduling smoother. This feature helps managers handle multiple loaded jobs by scheduling advance processing to avoid excessive concentration of resources during peak periods. Second, in contrast to traditional parallel machine scheduling models (e.g., Blazewicz, Dror, and Weglarz (1991) and Chen and Powell (1999)), the model described in this research focuses on the strictness of time constraints, making it particularly appropriate for air freight scheduling with high timeliness requirements. This implies that for logistics companies like Cainiao, the model may meet customers' demands for just-in-time delivery while also increasing the company's competitiveness through rapid response and flexible scheduling.

In practical applications, resource allocation should also be tailored to the specific operational context and constraints. If the primary objective is to complete all tasks before the 18:00 deadline, maintaining a makespan of 16:00—achievable with four or five workstations—would already suffice, ensuring all jobs are completed on time while avoiding unnecessary resource expenditure. Over-allocating resources, such as assigning seven or eight workstations, may lead to inefficiencies and increased costs. Thus, decision-makers should balance operational efficiency and resource utilization by conducting cost-benefit analyses to optimize resource allocation. This approach aligns with the BSP model's capability to minimize wasted capacity and ensure efficient resource management, particularly in time-sensitive air cargo operations.

Furthermore, the BSP model improves resource efficiency in logistics management by eliminating load imbalances and scheduling conflicts among workstations. The approach optimizes ULD loading time while allocating workstation resources efficiently, hence decreasing lost capacity caused by idle or overburdened resources. For actual logistics operations, the effective use of the BSP model demonstrates that by equitable job allocation and scheduling, logistics organizations can optimize workstation output under restricted resources, thereby making the whole process more efficient. This is especially useful for businesses aiming to manage demand and increase throughput while working with limited resources.

6.2. Limitation

Although the model proposed in this study has shown promising results in validation, it still faces some limitations when dealing with large-scale problems. First, in the Air Cargo Palletization Problem, the EPH algorithm's simplified placement strategy leads to a certain loss in space utilization. Specifically, the EPH algorithm prioritizes a default placement orientation and only considers rotation adjustments if the default placement fails. This simplification limits the full utilization of the ULD space, creating irregular gaps and reducing packing efficiency in large-scale cargo scenarios.

Moreover, while the MIP model can provide precise solutions for small datasets, its computation time increases significantly as the problem size grows, making it challenging to deliver timely solutions in logistics scenarios that require quick responses. This limitation restricts the MIP model's application in cases requiring high packing efficiency and rapid response. Future research could explore hybrid algorithms by combining global optimization and heuristic approaches. For example, integrating simulated annealing with EPH could optimize place-

ment strategies across the solution space while keeping computational complexity manageable, thereby enhancing the model's efficiency and solution quality in large-scale logistics contexts.

In addition to the existing limitations, another significant constraint of the proposed framework lies in its strictly sequential structure, which lacks a feedback mechanism between the Air Cargo Palletization and Build-up Scheduling stages. This absence of a feedback loop means that any inefficiencies or infeasible solutions identified during the scheduling stage cannot retroactively influence the packing decisions. While this design ensures simplicity and computational efficiency, it limits the framework's ability to dynamically adapt to changes or inconsistencies that arise downstream in the logistics process. For instance, packing decisions that maximize space utilization may inadvertently create bottlenecks or inefficiencies during build-up scheduling due to resource constraints or job dependencies.

Additionally, in both Air Cargo Palletization and Build-up Scheduling problems, the model currently assumes all job and cargo information is known before scheduling, without considering the dynamic changes and uncertainties present in real logistics operations. In practice, the arrival times and packing requirements of cargo may vary, and the current static scheduling model struggles to adapt to such fluctuations. This limits its ability to adjust packing sequences and resource allocation flexibly, potentially affecting overall scheduling efficiency. To address this, future research could incorporate a rolling horizon approach, enabling the model to update and optimize decisions iteratively as new information becomes available. For example, in the Air Cargo Palletization stage, a rolling horizon framework could periodically re-evaluate packing plans based on updated cargo arrival times and urgency levels, ensuring alignment with real-time constraints. Similarly, in the Build-up Scheduling stage, dynamic reallocation of jobs across workstations could mitigate resource imbalances and improve overall scheduling efficiency.

6.3. Conclusion

In response to Cainiao's logistics challenges, this study proposes a comprehensive loading plan that addresses the Air Cargo Palletization Problem (APP) and the Build-up Scheduling Problem (BSP) in two key stages. The first stage determines how each item is optimally packed into Unit Load Devices (ULDs), while the second stage establishes the precise timing for packing each ULD. This two-stage model achieves spatial and temporal optimization, enabling efficient, secure, and timely cargo loading and delivery.

In the first stage, Mixed Integer Programming and Extreme Point Heuristic methods are employed to solve the 3DBPP, which involves arranging variously shaped and sized items within limited ULD space while respecting different delivery deadlines. The MIP approach provides a precise algorithm that incorporates constraints such as weight capacity, stability, item orientation, and release and due dates. Validation demonstrates that the MIP model excels in small-scale problems, maximizing space utilization and balancing delays and ULD usage through penalty parameter adjustments. The EPH algorithm, by reducing the possible placement points of items within ULDs, avoids extensive search computations. While slightly lower in space utilization compared to MIP, EPH performs well in large-scale scenarios due to its quick solution generation, making it suitable for situations requiring rapid decision-making. From a management perspective, this phase offers a practical approach to selecting ULD types based on cargo dimension, adjusting delay penalty parameters and applying algorithms flexibly, which helps to strike a balance between improving loading efficiency and achieving on-time delivery, thus reducing operating costs and enhancing the competitiveness of enterprises in the global supply chain.

In the second phase, the BSP model addresses the loading scheduling problem, i.e., how to allocate jobs among multiple parallel workstations in a way that ensures that each ULD can be loaded within a strict time window. The model effectively optimizes workstation load balancing and avoids delays, especially when dealing with complex time-span scheduling, reducing the overall workstation working time and ensuring Cainiao's efficient operation while meeting the customer's need for just-in-time delivery. The BSP's time-sensitive scheduling also provides logistics managers with a framework to optimize workstation resource allocation, ensuring punctual deliveries and smoother operation workflows.

Building on the discussions of the two-stage framework and its applications, this study highlights the broader managerial and practical implications of integrating the Air Cargo Palletization Problem and the Build-up Scheduling Problem into a cohesive decision-making process. By addressing these challenges sequentially, the framework not only enhances operational efficiency but also provides actionable insights into how logistics operations can balance spatial optimization with temporal constraints. It provides valuable insights for managers aiming to optimize resource usage, reduce costs and improve service quality.

However, several limitations must be acknowledged, as they highlight areas for future research and improvement. A notable limitation of the framework is that its sequential structure lacks a feedback loop between the two stages. This restricts the framework's ability to dynamically adjust upstream decisions based on downstream inefficiencies. Additionally, the model's static nature assumes complete knowledge of job and cargo information prior to scheduling, making it less effective in handling dynamic changes and uncertainties inherent in practical scenarios. The models in the framework perform well in validation, but they have certain limitations in handling large-scale logistics problems. The simplified placement strategy in the EPH algorithm results in some loss of space utilization, and the MIP model's computational time increases significantly with problem size.

To address these challenges, future enhancements, such as the introduction of the iterative feedback loop, could further strengthen its applicability in complex and dynamic logistical settings, making it more suitable for real-world logistics environments where uncertainty and disruption are prevalent. Future research could explore combining dynamic adjustments and global optimization strategies to better address uncertainties in logistics operations. For example, incorporating a rolling horizon approach could achieve iterative updating of packaging and scheduling decisions. Additionally, hybrid approaches, such as combining simulated annealing with EPH, could enhance both solution quality and computational efficiency in large-scale logistics applications by optimizing placement globally while maintaining manageable complexity.

Overall, the proposed solution demonstrates high practical value for Cainiao's logistics operations, particularly in addressing the efficiency and flexibility required for cross-border e-commerce. This study also lays the groundwork for future research. With further optimization, the methods proposed here hold promise for broader application in larger and more complex logistics scenarios.

References

[1] Bombelli Alessandro et al. “From theORy to application: learning to optimize with Operations Research in an interactive way”. In: TU Delft OPEN Publishing, 2024. Chap. 9.

[2] Claudio Arbib and Fabrizio Marinelli. “Maximum lateness minimization in one-dimensional bin packing”. In: *Omega* 68 (2017), pp. 76–84.

[3] Claudio Arbib and Fabrizio Marinelli. “On cutting stock with due dates”. In: *Omega* 46 (2014), pp. 11–20.

[4] Oğuzhan Ahmet Arık. “Comparisons of metaheuristic algorithms for unrelated parallel machine weighted earliness/tardiness scheduling problems”. In: *Evolutionary Intelligence* 13.3 (2020), pp. 415–425.

[5] Jan Bank and Frank Werner. “Heuristic algorithms for unrelated parallel machine scheduling with a common due date, release dates, and linear earliness and tardiness penalties”. In: *Mathematical and computer modelling* 33.4-5 (2001), pp. 363–383.

[6] Julia A Bennell, Lai Soon Lee, and Chris N Potts. “A genetic algorithm for two-dimensional bin packing with due dates”. In: *International Journal of Production Economics* 145.2 (2013), pp. 547–560.

[7] Eberhard E Bischoff and MSW Ratcliff. “Issues in the development of approaches to container loading”. In: *Omega* 23.4 (1995), pp. 377–390.

[8] Jacek Blazewicz, Moshe Dror, and Jan Weglarz. “Mathematical programming formulations for machine scheduling: A survey”. In: *European Journal of Operational Research* 51.3 (1991), pp. 283–300.

[9] Felix Brandt and Stefan Nickel. “The air cargo load planning problem-a consolidated problem definition and literature review on related problems”. In: *European Journal of Operational Research* 275.2 (2019), pp. 399–410.

[10] Cainiao. *Global Logistics*. `https://www.cainiao.com/global-supply-chain.html?spm=a2d524.28498168.0.0.3c91ff05Lkwabv&from=subHeaderClick`. Accessed: 2024-03-25. 2024.

[11] Felix TS Chan et al. “Development of a decision support system for air-cargo pallets loading problem: A case study”. In: *Expert Systems with Applications* 31.3 (2006), pp. 472–485.

[12] Chin-Sheng Chen, Shen-Ming Lee, and QS Shen. “An analytical model for the container loading problem”. In: *European Journal of operational research* 80.1 (1995), pp. 68–76.

[13] Mingzhou Chen, Jiazhen Huo, and Yongrui Duan. “A hybrid biogeography-based optimization algorithm for three-dimensional bin size designing and packing problem”. In: *Computers & Industrial Engineering* 180 (2023), p. 109239.

[14] Zhi-Long Chen and Warren B Powell. “A column generation based decomposition algorithm for a parallel machine just-in-time scheduling problem”. In: *European Journal of Operational Research* 116.1 (1999), pp. 220–232.

[15] Zhi-Long Chen and Warren B Powell. “Solving parallel machine scheduling problems by column generation”. In: *INFORMS Journal on Computing* 11.1 (1999), pp. 78–94.

[16] Teodor Gabriel Crainic, Guido Perboli, and Roberto Tadei. “Extreme point-based heuristics for three-dimensional bin packing”. In: *Informs Journal on computing* 20.3 (2008), pp. 368–384.

[17] Simon Emde et al. “Scheduling personnel for the build-up of unit load devices at an air cargo terminal with limited space”. In: *OR Spectrum* 42.2 (2020), pp. 397–426.

[18] Stefano Fazi, Tom Van Woensel, and Jan C Fransoo. “A stochastic variable size bin packing problem with time constraints”. In: (2012).

[19] Paul C Gilmore and Ralph E Gomory. “A linear programming approach to the cutting stock problem—Part II”. In: *Operations research* 11.6 (1963), pp. 863–888.

[20] Dejan Gradišar and Miha Glavan. “Material Requirements Planning Using Variable-Sized Bin-Packing Problem Formulation with Due Date and Grouping Constraints”. In: *Processes* 8.10 (2020). ISSN: 2227-9717. URL: https://www.mdpi.com/2227-9717/8/10/1246.

[21] Fatma Gzara, Samir Elhedhli, and Burak C Yildiz. “The pallet loading problem: Three-dimensional bin packing with practical constraints”. In: *European Journal of Operational Research* 287.3 (2020), pp. 1062–1074.

[22] Thi Hong Ha Hai, Narameth Nananukul, et al. “Air cargo loading management systems for logistics forwarders”. In: (2016).

[23] International Air Transport Association. *IATA Annual Review 2023*. https://www.iata.org/contentassets/c81222d96c9a4e0bb4ff6ced0126f0bb/annual-review-2023.pdf. Accessed: 2024-03-25. 2023.

[24] International Air Transport Association. *The Value Of Air Cargo*. https://www.iata.org/contentassets/62bae061c05b429ea508cb0c49907c4c/air-cargo-brochure.pdf. Accessed: 2024-03-25. 2023.

[25] Leonardo Junqueira, Reinaldo Morabito, and Denise Sato Yamashita. “Three-dimensional container loading models with cargo stability and load bearing constraints”. In: *Computers & Operations Research* 39.1 (2012), pp. 74–85.

[26] J Kaabi et al. “Toward smart logistics: A new algorithm for a multi-objective 3D bin packing problem”. In: *Smart Cities Symposium 2018*. IET. 2018, pp. 1–5.

[27] AHG Rinnooy Kan. “Optimization and approximation in deterministic sequencing and scheduling: a survey”. In: *Annals of discrete mathematics*. Vol. 5. Elsevier, 1979, pp. 287–326.

[28] Chung-Yee Lee and Zhi-Long Chen. “Scheduling jobs and maintenance activities on parallel machines”. In: *Naval Research Logistics (NRL)* 47.2 (2000), pp. 145–165.

[29] Kunpeng Li, Appa Iyer Sivakumar, and Viswanath Kumar Ganesan. “Complexities and algorithms for synchronized scheduling of parallel machine assembly and air transportation in consumer electronics supply chain”. In: *European Journal of Operational Research* 187.2 (2008), pp. 442–455.

[30] Sabine Limbourg, Michaël Schyns, and Gilbert Laporte. “Automatic aircraft cargo load planning”. In: *Journal of the Operational Research Society* 63 (2012), pp. 1271–1283.

[31] Jin-Ling Lin, Chir-Ho Chang, and Jia-Yan Yang. “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.

[32] Qiang Liu et al. “Algorithms for the variable-sized bin packing problem with time windows”. In: *Computers & Industrial Engineering* 155 (2021), p. 107175.

[33] Andrea Lodi, Silvano Martello, and Daniele Vigo. “Recent advances on two-dimensional bin packing problems”. In: *Discrete Applied Mathematics* 123.1-3 (2002), pp. 379–396.

[34] Logicall. *Air Pallets Information-UNIT LOAD DEVICES*. Accessed: 2024-10-30. URL: `https://www.normanglobal.com/air-pallets-information/`.

[35] Fabrizio Marinelli and Andrea Pizzuti. “A Sequential Value Correction heuristic for a bi-objective two-dimensional bin-packing”. In: *Electronic Notes in Discrete Mathematics* 64 (2018), pp. 25–34.

[36] Silvano Martello and Paolo Toth. “Lower bounds and reduction procedures for the bin packing problem”. In: *Discrete applied mathematics* 28.1 (1990), pp. 59–70.

[37] Silvano Martello et al. “Algorithm 864: General and robot-packable variants of the three-dimensional bin packing problem”. In: *ACM Transactions on Mathematical Software (TOMS)* 33.1 (2007), 7–es.

[38] Ethel Mokotoff. “Parallel machine scheduling problems: A survey”. In: *Asia-Pacific Journal of Operational Research* 18.2 (2001), p. 193.

[39] Célia Paquay, Sabine Limbourg, and Michaël Schyns. “A tailored two-phase constructive heuristic for the three-dimensional Multiple Bin Size Bin Packing Problem with transportation constraints”. In: *European Journal of Operational Research* 267.1 (2018), pp. 52–64.

[40] Célia Paquay, Michael Schyns, and Sabine Limbourg. “A mixed integer programming formulation for the three-dimensional bin packing problem deriving from an air cargo application”. In: *International Transactions in Operational Research* 23.1-2 (2016), pp. 187–213.

[41] Sergey Polyakovskiy and Rym M’Hallah. “A hybrid feasibility constraints-guided search to the two-dimensional bin packing problem with due dates”. In: *European journal of operational research* 266.3 (2018), pp. 819–839.

[42] Sergey Polyakovskiy and Rym M’Hallah. “An intelligent framework to online bin packing in a just-in-time environment”. In: *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*. Springer. 2011, pp. 226–236.

[43] Sergey Polyakovskiy and Rym M’Hallah. “Just-in-time two-dimensional bin packing”. In: *Omega* 102 (2021), p. 102311.

[44] Harald Reinertsen and Thomas WM Vossen. “The one-dimensional cutting stock problem with due dates”. In: *European Journal of Operational Research* 201.3 (2010), pp. 701–711.

[45] Yasin Unlu and Scott J Mason. “Evaluation of mixed integer programming formulations for non-preemptive parallel machine scheduling problems”. In: *Computers & Industrial Engineering* 58.4 (2010), pp. 785–800.

[46] JM Van Den Akker, JA Hoogeveen, and Jules W van Kempen. “Parallel machine scheduling through column generation: Minimax objective functions”. In: *Algorithms–ESA 2006: 14th Annual European Symposium, Zurich, Switzerland, September 11-13, 2006. Proceedings 14*. Springer. 2006, pp. 648–659.

[47] Ying Yang et al. “Two-layer heuristic for the three-dimensional bin design and packing problem”. In: *Engineering Optimization* (2023), pp. 1–38.

[48] Abdolahad Noori Zehmakan. “Bin packing problem: Two approximation algorithms”. In: *arXiv preprint arXiv:1508.01376* (2015).

A

Data set for EPH

Table A.1: Dataset for EPH model

NO	Weight (kg)	Length (cm)	Width (cm)	Height (cm)	Release date	Due date
FRPALLET2405240003	192.94	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
FRPALLET2405240006	248.09	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
FRPALLET2405240009	98.08	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
FRPALLET2405240012	147.22	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
FRPALLET2405240020	205.75	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
FRPALLET2405240022	177.41	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
FRPALLET2405240024	83.49	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
FRPALLET2405240026	177.96	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
FRPALLET2405240027	93.111	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
FRPALLET2405240025	9.56	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
DEPALLET2405240001	134.318	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
DEPALLET2405240004	133.282	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
DEPALLET2405240010	184.822	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
DEPALLET2405240011	101.319	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
DEPALLET2405240013	151.296	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
DEPALLET2405240015	114.026	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
DEPALLET2405240008	102.343	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
NLPALLET2405240001	134.318	160	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
NLPALLET2405240002	133.282	160	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
NLPALLET2405240003	184.822	160	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
NLPALLET2405240004	172.68	160	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
NLPALLET2405240005	114.72	160	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
NLPALLET2405240006	189.76	160	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
FRPALLET2405240001	133.05	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
FRPALLET2405240002	175.74	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
FRPALLET2405240004	144.4	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
FRPALLET2405240005	169.25	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
FRPALLET2405240007	161.99	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
FRPALLET2405240010	127.47	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM

Table A.1: Dataset for EPH model

NO	Weight (kg)	Length (cm)	Width (cm)	Height (cm)	Release date	Due date
FRPALLET2405240011	235.72	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
FRPALLET2405240013	135.71	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
FRPALLET2405240014	116.68	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
DEPALLET2405240002	128.744	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
DEPALLET2405240003	145.472	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
DEPALLET2405240005	215.46	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
DEPALLET2405240006	243.851	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
DEPALLET2405240007	266.54	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
NLPALLET2405240007	237.07	160	120	100	2024/5/26 12:00:00 PM	2024/5/27 18:00:00 PM
NLPALLET2405240008	172.68	160	120	100	2024/5/26 12:00:00 PM	2024/5/27 18:00:00 PM
NLPALLET2405240009	114.72	160	120	100	2024/5/26 12:00:00 PM	2024/5/27 18:00:00 PM
NLPALLET2405240010	189.76	160	120	100	2024/5/26 12:00:00 PM	2024/5/27 18:00:00 PM
NLPALLET2405240011	84.58	160	120	100	2024/5/26 12:00:00 PM	2024/5/27 18:00:00 PM
FRPALLET2405270006	139.29	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
FRPALLET2405270009	237.07	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
FRPALLET2405270013	172.68	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
FRPALLET2405270016	114.72	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
FRPALLET2405270018	189.76	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
FRPALLET2405270019	84.58	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
FRPALLET2405270020	112.85	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
FRPALLET2405270021	156.12	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
FRPALLET2405270022	141.22	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
FRPALLET2405270024	17.27	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
DEPALLET2405270006	216.046	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
DEPALLET2405270008	149.18	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
DEPALLET2405270011	93.953	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
NLPALLET2405270001	237.07	160	120	100	2024/5/27 12:00:00 PM	2024/5/28 18:00:00 PM
NLPALLET2405270002	172.68	160	120	100	2024/5/27 12:00:00 PM	2024/5/28 18:00:00 PM
NLPALLET2405270003	114.72	160	120	100	2024/5/27 12:00:00 PM	2024/5/28 18:00:00 PM

Table A.1: Dataset for EPH model

NO	Weight (kg)	Length (cm)	Width (cm)	Height (cm)	Release date	Due date
NLPALLET2405270004	189.76	160	120	100	2024/5/27 12:00:00 PM	2024/5/28 18:00:00 PM
NLPALLET2405270005	84.58	160	120	100	2024/5/27 12:00:00 PM	2024/5/28 18:00:00 PM
FRPALLET2405270001	192.94	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
FRPALLET2405270002	248.09	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
FRPALLET2405270003	98.08	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
FRPALLET2405270004	147.22	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
FRPALLET2405270005	205.75	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
FRPALLET2405270007	177.41	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
FRPALLET2405270008	83.49	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
FRPALLET2405270010	177.96	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
FRPALLET2405270011	93.111	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
FRPALLET2405270012	9.56	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
DEPALLET2405270001	134.318	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
DEPALLET2405270002	133.282	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
DEPALLET2405270003	184.822	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
DEPALLET2405270004	101.319	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
DEPALLET2405270005	151.296	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
DEPALLET2405270009	114.026	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
DEPALLET2405270007	102.343	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
NLPALLET2405270006	134.318	160	120	100	2024/5/27 12:00:00 PM	2024/5/28 18:00:00 PM
NLPALLET2405270007	133.282	160	120	100	2024/5/27 12:00:00 PM	2024/5/28 18:00:00 PM
NLPALLET2405270008	184.822	160	120	100	2024/5/27 12:00:00 PM	2024/5/28 18:00:00 PM
NLPALLET2405270009	172.68	160	120	100	2024/5/27 12:00:00 PM	2024/5/28 18:00:00 PM
NLPALLET2405270010	114.72	160	120	100	2024/5/27 12:00:00 PM	2024/5/28 18:00:00 PM
NLPALLET2405270011	189.76	160	120	100	2024/5/27 12:00:00 PM	2024/5/28 18:00:00 PM
FRPALLET2405240015	192.94	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
FRPALLET2405240016	248.09	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
FRPALLET2405240017	98.08	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
FRPALLET2405240018	147.22	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM

Table A.1: Dataset for EPH model

NO	Weight (kg)	Length (cm)	Width (cm)	Height (cm)	Release date	Due date
FRPALLET2405240019	205.75	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
FRPALLET2405240022	177.41	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
FRPALLET2405240021	83.49	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
FRPALLET2405240023	177.96	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
FRPALLET2405240028	93.111	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
FRPALLET2405240029	9.56	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
DEPALLET2405240009	134.318	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
DEPALLET2405240012	133.282	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
DEPALLET2405240014	184.822	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
DEPALLET2405240016	101.319	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
DEPALLET2405240017	151.296	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
DEPALLET2405240018	114.026	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
DEPALLET2405240019	102.343	140	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
NLPALLET2405240012	134.318	160	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
NLPALLET2405240013	133.282	160	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
NLPALLET2405240014	184.822	160	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
NLPALLET2405240015	172.68	160	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
NLPALLET2405240016	114.72	160	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
NLPALLET2405240017	189.76	160	120	100	2024/5/27 12:00:00 PM	2024/5/27 18:00:00 PM
FRPALLET2405270014	139.29	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
FRPALLET2405270015	237.07	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
FRPALLET2405270017	172.68	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
FRPALLET2405270023	114.72	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
FRPALLET2405270025	189.76	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
FRPALLET2405270026	84.58	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
FRPALLET2405270027	112.85	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
FRPALLET2405270028	156.12	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
FRPALLET2405270029	141.22	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
FRPALLET2405270030	17.27	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM

Table A.1: Dataset for EPH model

NO	Weight (kg)	Length (cm)	Width (cm)	Height (cm)	Release date	Due date
DEPALLET2405270010	216.046	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
DEPALLET2405270012	149.18	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
DEPALLET2405270013	93.953	140	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
NLPALLET2405270012	237.07	160	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
NLPALLET2405270013	172.68	160	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
NLPALLET2405270014	114.72	160	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
NLPALLET2405270015	189.76	160	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
NLPALLET2405270016	84.58	160	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
NLPALLET2405270017	84.58	160	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM
NLPALLET2405270018	84.58	160	120	100	2024/5/28 12:00:00 PM	2024/5/28 18:00:00 PM

B

Loading results for EPH

Table B.1: Loading results for all used ULDs by using EPH

ULD	Loaded items	Allowable starting time	Duration	Earliest completion time	Earliest due date	Cost
AMA1	NLPALLET2405240007, NLPALLET2405240008, NLPALLET2405240009, NLPALLET2405240010, NLPALLET2405240011, FRPALLET2405240003	2024-05-27 12:00:00	1.5	2024-05-27 13:30:00	2024-05-27 18:00:00	347.11
AMA2	FRPALLET2405240006, FRPALLET2405240009, FRPALLET2405240012, FRPALLET2405240020, FRPALLET2405240022, FRPALLET2405240024, FRPALLET2405240026, FRPALLET2405240027	2024-05-27 12:00:00	2	2024-05-27 14:00:00	2024-05-27 18:00:00	430.89
AMA3	FRPALLET2405240025, DEPALLET2405240001, DEPALLET2405240004, DEPALLET2405240010, DEPALLET2405240011, DEPALLET2405240013, DEPALLET2405240015, DEPALLET2405240008	2024-05-27 12:00:00	2	2024-05-27 14:00:00	2024-05-27 18:00:00	325.84
AMA4	NLPALLET2405240001, NLPALLET2405240002, NLPALLET2405240003, NLPALLET2405240004, NLPALLET2405240005, NLPALLET2405240006	2024-05-27 12:00:00	1.5	2024-05-27 13:30:00	2024-05-27 18:00:00	325.35
AMA5	FRPALLET2405240001, FRPALLET2405240002, FRPALLET2405240004, FRPALLET2405240005, FRPALLET2405240007, FRPALLET2405240010, FRPALLET2405240011, FRPALLET2405240013	2024-05-27 12:00:00	2	2024-05-27 14:00:00	2024-05-27 18:00:00	449.17
AMA6	FRPALLET2405240014, DEPALLET2405240002, DEPALLET2405240003, DEPALLET2405240005, DEPALLET2405240006, DEPALLET2405240007, FRPALLET2405240015, FRPALLET2405240016	2024-05-27 12:00:00	2	2024-05-27 14:00:00	2024-05-27 18:00:00	545.22
AMA7	FRPALLET2405240017, FRPALLET2405240018, FRPALLET2405240019, FRPALLET2405240022, FRPALLET2405240021, FRPALLET2405240023, FRPALLET2405240028, FRPALLET2405240029	2024-05-27 12:00:00	2	2024-05-27 14:00:00	2024-05-27 18:00:00	347.40

Table B.1: Loading results for all used ULDs by using EPH

ULD	Loaded items	Allowable starting time	Duration	Earliest completion time	Earliest due date	Cost
AMA8	DEPALLET2405240009, DEPALLET2405240012, DEPALLET2405240014, DEPALLET2405240016, DEPALLET2405240017, DEPALLET2405240018, DEPALLET2405240019, NLPALLET2405240012	2024-05-27 12:00:00	2	2024-05-27 14:00:00	2024-05-27 18:00:00	369.50
AMA9	NLPALLET2405240013, NLPALLET2405240014, NLPALLET2405240015, NLPALLET2405240016, NLPALLET2405240017, NLPALLET2405270001	2024-05-27 12:00:00	1.5	2024-05-27 13:30:00	2024-05-27 18:00:00	361.32
AMA10	NLPALLET2405270002, NLPALLET2405270003, NLPALLET2405270004, NLPALLET2405270005, NLPALLET2405270006, NLPALLET2405270007	2024-05-27 12:00:00	1.5	2024-05-27 13:30:00	2024-05-28 18:00:00	290.27
AMA11	NLPALLET2405270008, NLPALLET2405270009, NLPALLET2405270010, NLPALLET2405270011, FRPALLET2405270006, FRPALLET2405270009, FRPALLET2405270013, FRPALLET2405270016	2024-05-28 12:00:00	2	2024-05-28 14:00:00	2024-05-28 18:00:00	464.01
AMA12	FRPALLET2405270018, FRPALLET2405270019, FRPALLET2405270020, FRPALLET2405270021, FRPALLET2405270022, FRPALLET2405270024, DEPALLET2405270006, DEPALLET2405270008	2024-05-28 12:00:00	2	2024-05-28 14:00:00	2024-05-28 18:00:00	373.46
AMA13	DEPALLET2405270011, FRPALLET2405270001, FRPALLET2405270002, FRPALLET2405270003, FRPALLET2405270004, FRPALLET2405270005, FRPALLET2405270007, FRPALLET2405270008	2024-05-28 12:00:00	2	2024-05-28 14:00:00	2024-05-28 18:00:00	436.43
AMA14	FRPALLET2405270010, FRPALLET2405270011, FRPALLET2405270012, DEPALLET2405270001, DEPALLET2405270002, DEPALLET2405270003, DEPALLET2405270004, DEPALLET2405270005	2024-05-28 12:00:00	2	2024-05-28 14:00:00	2024-05-28 18:00:00	344.98

Table B.1: Loading results for all used ULDs by using EPH

ULD	Loaded items	Allowable starting time	Duration	Earliest completion time	Earliest due date	Cost
AMA15	DEPALLET2405270009, DEPALLET2405270007, FRPALLET2405270014, FRPALLET2405270015, FRPALLET2405270017, FRPALLET2405270023, FRPALLET2405270025, FRPALLET2405270026	2024-05-28 12:00:00	2	2024-05-28 14:00:00	2024-05-28 18:00:00	404.06
AMA16	FRPALLET2405270027, FRPALLET2405270028, FRPALLET2405270029, FRPALLET2405270030, DEPALLET2405270010, DEPALLET2405270012, DEPALLET2405270013, NLPALLET2405270012	2024-05-28 12:00:00	2	2024-05-28 14:00:00	2024-05-28 18:00:00	393.30
AAP1	NLPALLET2405270013, NLPALLET2405270014, NLPALLET2405270015	2024-05-28 12:00:00	0.75	2024-05-28 12:45:00	2024-05-28 18:00:00	167.01
AAP2	NLPALLET2405270016, NLPALLET2405270017, NLPALLET2405270018	2024-05-28 12:00:00	0.75	2024-05-28 12:45:00	2024-05-28 18:00:00	88.81