Offshore Consolidation Centre for Air Cargo Operations: Model and Solution Methods

Xin Chu
Department of Industrial Systems Engineering and Management
National University of Singapore, Republic of Singapore
chu.xin@u.nus.edu

Yuan Wang †
School of Business
Singapore University of Social Sciences, Republic of Singapore
wangyuan@suss.edu.sg

Yun Hui Lin
Institute of High Performance Computing (IHPC)
Agency for Science, Technology and Research (A*STAR)
1 Fusionopolis Way, #16-16 Connexis, Singapore 138632, Republic of Singapore
liny@ihpc.a-star.edu.sg

Ek Peng Chew
Department of Industrial Systems Engineering and Management
National University of Singapore, Republic of Singapore
isecep@nus.edu.sg

† Corresponding author

Offshore Consolidation Centre for Air Cargo Operations: Model and Solution Methods

Abstract

Recent years have witnessed a surge in demand for air cargo transportation. Air cargo terminals initially built for handling the former demand volume start struggling with low operational efficiency and high congestion due to limited capacity. The industry has explored methods such as building new terminals and renting warehouses to stay competitive in the market. However, these methods may not be practical in many scenarios due to economic reasons and the scarcity of airport land. As an alternative method, this paper investigates an innovative operational paradigm with the concept of Offshore Consolidation Centre (OCC) to deal with the imbalance between the surging demand and the limited capacity of air cargo terminals. We redesign air cargo terminal operations with the existence of OCC and formulate the problem as a mixed-integer linear program (MILP). We then propose a two-stage alternating algorithm embedded with a tailored metaheuristic to analyse the validity of our model and algorithm. Experiment results reveal that our alternating algorithm has the best performance compared with the solver and benchmark policies adopted in industrial practice. Moreover, the comparison with air cargo terminals without OCC demonstrates that OCC can reduce delays, cargo waiting time, and traffic flow, by a large margin.

Key words: air cargo; Offshore Consolidation Centre; terminal operation; alternating algorithm

1. Introduction

1.1. Background

Air cargo transportation has become one of the essential primary services for the national economy and social development due to its high efficiency. As an indispensable part of air transportation, freight is hailed as 'an economic development engine' and has exhibited a symbiotic relationship with economic growth (Kasarda & Green, 2005). According to the International Air Transport Association, air cargoes serve 35% of the

global trade value but only represent less than 1% of trade volume. From the 1990s to 2010s, the air cargo volume grew dramatically, with an average annual growth rate of more than 25%[1]. The annual value of goods transported by air cargoes exceeded 6.7 trillion US dollars in 2020.

The air cargo terminal plays a vital role in the industry, and one of its most critical functions is to receive the local cargoes that are delivered to the terminal by trucks. As shown in Fig 1, typical operations include the processes of cargo unloading, cargo inspection (e.g., weighing and tagging), cargo waiting, unit load device (ULD) build-up, and storage.

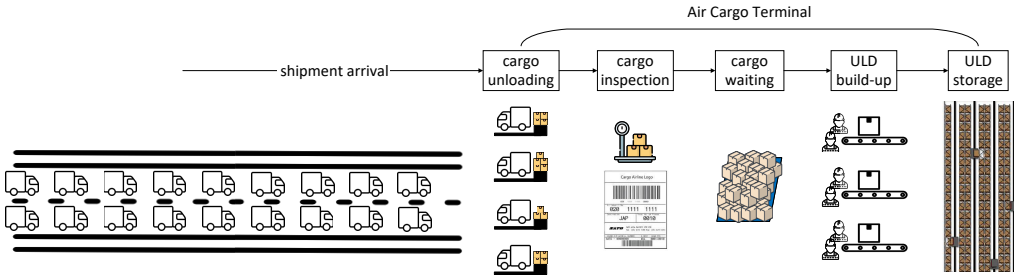


Figure 1: Traditional air cargo terminal operation processes

However, air cargo terminals have been reported to suffer from heavy congestion[2]. Above all, the cargo unloading and ULD build-up are the two significant steps that cause congestion in operation. Although the cargo unloading time is mainly within a half-hour, it may take up to 36 to 40 hours for trucks to enter terminal unloading bays[2]. With the limited space in the unloading bay of air cargo terminals, trucks have to queue up on the main road, which tremendously intensifies the congestion and deteriorates traffic conditions around the airport. Meanwhile, the current practice for the cargo unloading process adopts the 'first-come-first-serve' (FCFS) rule without anticipating the subsequent process. As a result, there is a lack of coordination between the cargo unloading and ULD build-up processes, leading to immense difficulty in the build-up schedule. In fact, most of the loose cargoes are to be built up into a ULD for freight transport, predetermined by the destination of each freighter. A ULD may carry several cargoes. Before the last cargo arrives at the terminal and completes its unloading process, all other cargoes that are assigned to the same ULD have to wait inside the terminal, which is one of the primary

[1]IATA. (2015). The Value of Air Cargo[Brochure].
https://www.iata.org/contentassets/4d3961c878894c8a8725278607d8ad52/air-cargo-brochure.pdf
[2]Damian Brett. 'Air cargo capacity continues to climb but so do rates' Air Cargo News, May 5th, 2020.

sources of terminal congestion and workforce shortage, especially during peak hours. Both issues will eventually lead to ULD delays (ULD missing flights), signalling the need for designing coordination schemes for the cargo unloading and ULD build-up processes.

In addition to the lack of coordination, another reason for the congestion is the growing demand for air cargo services. To satisfy the surge of online shopping and overseas business, airports across the United States are constructing new air cargo terminals. For example, Greenville-Spartanburg International Airport opened a 110,000-square-foot facility to increase its domestic and international air cargo operations in 2020[3]. FedEx has opened a 251,000-square-foot sorting facility in Ontario International Airport. Amazon is completing its new 798,000 square-foot centre in Cincinnati/Northern Kentucky International Airport[4]. Europe's airports also suffer from severe congestion due to surging demand. An air services provider at Amsterdam Airport Schiphol received a 40% increase in demand last year and leased a 10,000 square-meter facility to stem the chaos temporarily[5]. Swissport has leased warehouses at Vienna Airport to expand its air cargo business[6]. In a nutshell, these airports and companies dealt with the congestion problem by constructing new facilities or leasing space inside the airport. However, such a solution could lead to high costs and sometimes is not practical due to the space limitation of the airport. Besides, constructing new facilities or leasing space inside the airport can hardly mitigate traffic congestion around/outside the airport. All local trucks are still rushing to the airport, and the traffic condition is still deteriorating.

Furthermore, COVID-19 has extremely accelerated the demand growth for air cargo services since these services play a lifeline of vital supplies during the lockdowns. This, however, further leads to severe supply chain congestion in the post-pandemic era. The overall revenue of the air cargo industry in 2020 represented an increase of 10-15% compared to pre-pandemic levels, and the demands for air cargo in 2021 and 2022 are expected to exceed pre-pandemic levels by 8% and 13%, respectively[7]. Changi Airport in Singapore has

[3]Joe Petrie. 'Built to Move—Airports prepare for a strong cargo future by addressing their facilities now' Aviation Pros, May 14th, 2020.

[4]Keith Schneider. 'Air Cargo Construction Is Booming, Thanks to Amazon' The New York Times, Jan 12th, 2021.

[5]Alex Lennane. '"It's a juggling act" say handlers as cargo congestion swamps Europe's airports' The Loadstar, Feb 15th, 2021.

[6]Swissport International Ltd. 'Swissport Further Expands its Air Cargo Business at Vienna Airport with a Brand New Cargo Terminal' Aviation Pros, Mar 22th, 2021.

[7]IATA. 'Key to Air Cargo Resilience Post Pandemic: Cooperation, Safety, Sustainability, Modernization' 14th World Cargo Symposium, Oct 12th, 2021.

more than doubled its weekly cargo frequencies and increased cargo connectivity by over 40% to 70 cities. To support such demand surges, terminal operators have been adapting to a new normal of cargo operating procedures[8]. Concisely, increasing the capacity alone may not be sufficient to overcome the emerging challenge. Instead, the key lies in enhancing both the capacity and operational efficiency of the entire industrial system in tandem. This dual approach is essential not only for addressing current congestion concerns but also for establishing resilience against potential future crises, be it a pandemic or other emergencies.

1.2. Literature Review

In the last two decades, the research on air cargo services has been gaining increasing attention. However, most of the research focused on the airline's perspective and investigated revenue management, fleet routing, and flight scheduling according to Feng et al. (2015). Only a few studies considered the air cargo terminal operation, which consists of fundamental decision-making problems such as truck scheduling and cargo/ULD processing.

Earlier works concentrated on truck arrival and cargo unloading management for truck scheduling problems and aimed to minimise the waiting time. Hall (2001) first constructed a queueing system to describe truck random arrivals, while Selinka et al. (2016) analysed a complex queueing system with two classes of trucks, multiple handling facilities, and state-dependent routing. Besides the intuitive queueing method, Ou et al. (2010) applied truck cross-docking strategies and formulated a two-stage integer program with the purpose of minimising the total handling and storage cost. Road feeder service is another research area related to air cargo land delivery. However, most papers in this field focus on solving routing and scheduling problems for periodic tasks, such as weekly or monthly operations, as demonstrated in studies like Bartodziej et al. (2009) and Derigs et al. (2011). These papers aimed to design efficient routes and trips considering specific time window constraints.

On the other hand, the cargo/ULD processing is typically the most time-consuming and labour-intensive stage in the handling agent's overall process chain according to Brandt & Nickel (2019). Existing research papers related to cargo/ULD processing mostly

[8]Changi Airport Group. 'How air cargo is adapting to the new normal' June, 2020.

focus on manpower planning, build-up process scheduling, and automated guided vehicle routing. Yan et al. (2006), Yan et al. (2008a), Yan et al. (2008b) conducted a series of studies focusing on the manpower supply planning for air cargo terminals, utilising real operational data from a Taiwan air cargo terminal to validate their models. Rong & Grunow (2009) formulated an integrated mixed-integer linear programming model to minimise manpower costs in ULD build-up and break-down processes. Emde et al. (2020) addressed the problem of scheduling the ULD build-up process to minimise the peak workforce, considering limited space, personnel constraints, and planning horizon. Lee et al. (2006), Lau & Zhao (2006), and Xu et al. (2014) aimed to minimise waiting time or maximise resource utilisation within automated air cargo handling systems through approaches like simulation analysis, heuristic approach, and flow allocation routing strategy, respectively.

Indeed, truck scheduling problems and cargo processing problems have been separately investigated, but there is a lack of research on their integration (Feng et al., 2015). Defining and solving integrated decision-making problems can not only bridge the current research gaps but also introduce coordination that can significantly enhance the efficiency of the air cargo land delivery process.

Consolidation centre is one of the most frequently implemented initiatives to alleviate the challenges in freight logistics according to Browne et al. (2005). An Urban Consolidation Centre (UCC) is a facility located near the city centre that serves as a hub for receiving goods from outside the city, and then consolidating and transporting to the customers within the city. The primary goal is to consolidate goods movement to optimise deliveries and reduce transportation costs. Furthermore, by integrating with appropriate vehicles, a UCC can help alleviate congestion issues and improve the efficiency of goods transportation Nordtømme et al. (2015). Considering the congestion issues and operational inefficiencies in air cargo land delivery operations, exploring the insights from UCC could potentially provide innovative solutions. Therefore, a comprehensive review of UCC and its expansion and implementation is necessary.

Allen et al. (2012) provided a comprehensive review of 114 UCC schemes from 17 countries between 2006 and 2010 in Europe. The paper discusses critical organisational, operational, and financial aspects of UCC success and reviews the traffic and environmental impacts of UCC trials and operational schemes. Jacyna (2013) presented research results on

the city logistics system in Poland and focused on the use of consolidation centres to address logistics service challenges in urban agglomerations. Van Heeswijk et al. (2019) examined sustainable business models and administrative policies supporting UCCs. 1,458 schemes were tested in the context of Copenhagen and the findings propose various propositions to create favourable conditions for UCCs, increasing their chances of long-term success.

Although there were sufficient explorations in the industry, the complex consolidation operations inside UCC only received limited attention in the research community. Crainic et al. (2004) proposed that consolidation is the primary concern of UCC since it largely influences the design and location of UCC. However, the focus of this study was on addressing planning and operational issues and developing models for effective control of freight movements within city centres, rather than the consolidation operation within UCC. Verlinde et al. (2012) demonstrated the benefits of consolidation in classic urban logistics. However, the consolidation process in UCC was only filling the delivery vehicle to its limit in this article, which is not a decent strategy. Besides, it has been argued that the lack of an efficient consolidation strategy is one of the main reasons for unsuccessful UCC in Vahrenkamp & Berlin (2016).

In addition to the absence of operational consolidation strategies, typical UCC problems often overlook the consideration of follow-up tasks, which results in limited application potential and a disconnection between the upstream and downstream of the supply chain. As a result, UCC implementation or its extension remains an isolated initiative, leading to unsatisfactory economic benefits. In fact, according to Dablanc et al. (2011), the majority (96%) of UCCs in Europe did not survive in practice.

To successfully implement a consolidation centre in air cargo land delivery operations, it is crucial to consider the follow-up tasks and develop a customised operational consolidation strategy tailored to the specific requirements of air cargo operations. By addressing these challenges and developing an integrated approach, the potential benefits of a consolidation centre can be fully realised.

1.3. Contribution

Positioning within the literature of UCC and air cargo terminal operations, our contributions are as follows.

- *Problem and model.* We propose the Offshore Consolidation Centre (OCC) to deal with the imbalance between the increasing demand for air cargo services and the limited

capacity of existing air cargo terminals and transfer the traffic pressure around the airport. It integrates truck scheduling and cargo processing problems, introducing coordination to the entire industrial system and filling the research gap in this field. We construct a two-stage system for the whole operation process and formulate a MILP model. It considers the subsequent process (ULD build-up at the terminal) at the previous stage (cargo consolidation at OCC), which coordinates the follow-up tasks with traditional UCC problems and analyses the operational activities in OCC. It complements the research on consolidation activities and expands the practicality of UCC.

- *Solution approaches.* To make our framework practically implementable, we design a two-stage alternating algorithm. It first generates an initial solution for the first stage using a greedy algorithm and then solves the second stage with a tailored Genetic Algorithm (GA) based on the solution of the first stage. To improve the performance of GA, we implement several enhancement strategies, such as a local search-based crossover method and a tailored unimprovement factor. To avoid the local optimum, we employ Gurobi to obtain the optimal solution for the first stage based on the solution from GA. Finally, we use GA and Gurobi alternatively to achieve better solutions until the time or iteration limit.

- *Managerial implications.* We test OCC and our proposed alternating algorithm in the actual size cases. The results show that our alternating algorithm consistently produces the largest earliness (remaining time) and the least tardiness (delayed ULDs) in every case compared to the solver and benchmarks (the first-come-first-serve and first-due-first-serve rule) within a time limit. We also compare OCC and the current policy (air cargo terminal operations without OCC) to verify the practicality of OCC. The results show that OCC reduces the traffic flow to the airport, eliminates congestion at the air cargo terminal and decreases the ULD delays and cargo waiting time (average reduction of waiting time can go up to 69%). Besides, we investigate the effect of OCC under different scenarios by changing arrival patterns. Poisson distribution is adopted to simulate stochastic arrivals. The triangular distribution is adopted to present the situation where most truck drivers prefer to arrive late. An extreme case where all trucks arrive at time 0 indicates backlog condition. All the experiments demonstrate the superiority of our OCC model compared to benchmarks.

The remainder is organised as follows: Section 2 presents the problem description and mathematical formulation of the operational processes with OCC. Section 4 introduces a two-stage alternating algorithm to solve the problem in actual practice. The numerical experiments are conducted in Section 5. Section 6 summarises our conclusions and offers suggestions for further research.

2. Problem Definition and Description

2.1. Problem Definition

Inspired by the air cargo terminal operations and Urban Consolidation Centre, we propose the Offshore Consolidation Centre (OCC) to mitigate the imbalance between the terminal capacity and the growing demand. OCC is designed to receive shipments carried by external trucks at a distance from the airport. As shown in Fig 2, the new processes with OCC include arrival shipment received at OCC, cargo inspection and consolidation at OCC, cargo transferred from OCC to the terminal, cargo unloading, cargo waiting, ULD build-up, and storage at the terminal. Inside OCC, cargoes are required to be rearranged and transported to the terminal according to a transfer policy. Meanwhile, the ULD build-up process at the terminal is scheduled with the transfer policy at the same time.

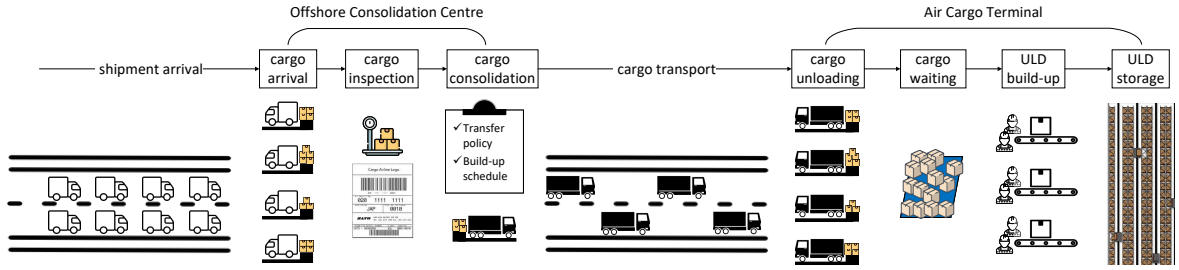


Figure 2: Air cargo terminal operation processes with OCC

In essence, the adoption of OCC allows for increasing capacity and enhancing the efficiency of the entire operation. On the one hand, OCC enlarges the capacity for sorting incoming cargoes before they enter the terminal, enabling us to schedule the arrival of cargoes at the terminal. More significantly, transferring the traffic pressure around the airport to OCC helps control the traffic flow to the airport and diminish the congestion around the airport, thereby reinforcing the efficiency of the cargo unloading process. On the other hand, the ULD build-up policy is determined simultaneously with the cargo transferring and unloading policy in the early stage of cargo consolidation in OCC so that

we can concatenate all operation processes and resolve the two bottlenecks synchronously. Based on these advantages, it is possible for us to mitigate the operation burden arising from the surging demand and take the initiative to reduce terminal congestion, leveraging a well-designed terminal operation strategy with OCC.

Finally, we point out that OCC can be established as a temporary or permanent infrastructure near the airport. Temporary OCC is able to help the government deal with emergencies at special times, such as the COVID-19 pandemic that does not allow for huge traffic congestion at the airport. Permanent OCC requires higher investment, but it helps terminal operators enhance resiliency and strengthen supply chain management by controlling the traffic at the airport. Although OCC has not been widely employed thus far, this paper aims to investigate, if this novel operational paradigm is put into practice, how much benefit it can bring to operation efficiency. Specifically, throughout our modelling and computational study in the following sections, we demonstrate that OCC can effectively mitigate the congestion around the airport and accommodate the surging demand for air cargo services. Besides, we manifest that OCC can perform as a practical and competent scheme for the authority to implement for the next pandemic or emergency.

2.2. Problem Description

The air cargo terminal operations with OCC for loose cargoes which are required to be built up into ULDs include the following processes.

- Cargo arrives at OCC. Each cargo $i \in I$ (I is the set of cargoes) is delivered by an external truck $g \in G$ (G is the set of external trucks) and arrives at OCC at arrival time A_g.

- Cargo consolidation at OCC. Each cargo needs to be consolidated and reassigned to a transport $s \in S$ (S is the set of truck transport from OCC to the terminal). After the inspection time T_0 and consolidation time T_1, each transport will depart at departure time d_s.

- Cargo unloading at the terminal. After the transportation time T_2 and unloading time T_3, each cargo enters the terminal at time u_i and waits for the build-up process.

- ULD build-up. The build-up process of ULD $j \in J$ (J is the set of ULDs) on a workstation $w \in W$ (W is the set of workstations) is completed at time c_j. The unloading completion time of the related cargoes u_i is the connection between the two stages.

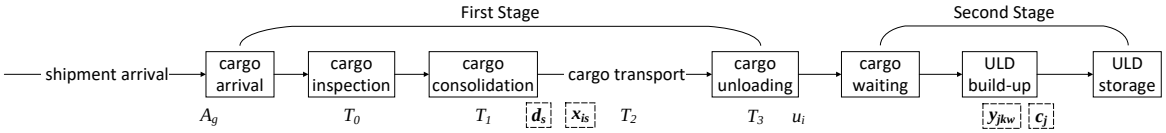


Figure 3: Two-stage system of the processes with OCC

The whole process can be regarded as a two-stage system, shown in Fig 3. The first stage includes cargo arrival, cargo inspection and consolidation, cargo transport by internal trucks, and unloading processes. Because the transportation time T_2 is assumed to be the same among all transports, as well as the inspection time T_0, consolidation time T_1, and unloading time T_3, we can assign the transport i and transport $i + B$ using the same unloading bay, where B is the number of unloading bays. Therefore, we focus on the assignment between cargoes and transports and use binary decision variable x_{is}, where x_{is} takes the value 1 if cargo i is transferred from OCC to the terminal by transport s, and 0 otherwise. The second stage includes the ULD build-up process. It is a parallel machine scheduling problem. We use binary decision variable y_{jkw}, where y_{jkw} takes value 1 if the build-up process of ULD j is directly followed by ULD k on workstation w, and 0 otherwise.

3. Mathematical Formulation

With the main notations in Table 1, we propose the following mathematical model for optimising the above two-stage system.

Table 1: Notations

Indices:	
I	set of cargoes $\{1,2,...,n\}$
J, K	set of ULDs $\{0,1,2,...,m\}$, where $'0'$ indicates dummy job
G	set of arrival external trucks $\{1,2,...,g\}$
S	set of cargo transports from OCC to the terminal by internal trucks $\{1,2,...,s\}$

W	set of workstations {1,2,...,w}

Parameters:

L	internal truck loading capacity
B	the number of unloading bays
T_0	processing time for cargo inspection at OCC
T_1	processing time for cargo consolidation at OCC
T_2	transportation time from OCC to the terminal
T_3	processing time for unloading a truck at the terminal
P_j	processing time for building up ULD j
D_j	due date of ULD j according to the flight schedule
A_g	arrival time for external truck g at OCC
a_{ij}	the assignment between cargoes and ULDs ($a_{ij} = 1$, if cargo i will be built up into ULD j; $a_{ij} = 0$, otherwise)
b_{ig}	the assignment between cargoes and external trucks ($b_{ig} = 1$, if cargo i is delivered to OCC by external truck g; $b_{ig} = 0$, otherwise)
M	a sufficiently large number
Q	unit penalty cost for ULD tardiness

Intermediary Variables:

u_i	unloading completion time for cargo i at the terminal

Decision Variables:

d_s	departure time for the transport s from OCC to the terminal
c_j	completion time for the build-up process of ULD j
x_{is}	$x_{is} = 1$, if cargo i is transferred from OCC to the terminal by the transport s; $x_{is} = 0$, otherwise
z_s	$z_s = 1$, if the transport s is executed; $z_s = 0$, otherwise
y_{jkw}	$y_{jkw} = 1$, if build-up process for ULD j is directly followed by ULD k on workstation w; $y_{jkw} = 0$, otherwise

$$\max \quad \sum_{j>0}[(D_j - c_j)^+ - Q(c_j - D_j)^+] \tag{1}$$

$$\text{s.t. } d_{s-B} + T_3 \leq d_s \qquad \forall s > B \tag{2}$$

$$d_{s-1} \leq d_s \qquad \forall s > 1 \tag{3}$$

$$z_{s-1} \geq z_s \qquad \forall s > 1 \qquad (4)$$

$$b_{ig} A_g + T_0 + T_1 \leq \sum_s x_{is} d_s \qquad \forall i \in I \qquad (5)$$

$$\sum_s x_{is} d_s + T_2 + T_3 = u_i \qquad \forall i \in I \qquad (6)$$

$$\sum_s x_{is} = 1 \qquad \forall i \in I \qquad (7)$$

$$x_{is} \leq z_s \qquad \forall i \in I, s \in S \qquad (8)$$

$$\sum_i x_{is} \leq z_s L \qquad \forall s \in S \qquad (9)$$

$$\sum_{k \neq j} \sum_w y_{jkw} = 1 \qquad \forall j > 0 \qquad (10)$$

$$\sum_{k \neq j} y_{jkw} = \sum_{k \neq j} y_{kjw} \qquad \forall w, j > 0 \qquad (11)$$

$$\sum_{k>0} y_{0kw} \leq 1 \qquad \forall w \in W \qquad (12)$$

$$c_j \geq c_k + P_j - M(1 - y_{kjw}) \qquad \forall w, j, k > 0, j \neq k \qquad (13)$$

$$c_j \geq a_{ij} u_i + P_j \qquad \forall i \in I, j > 0 \qquad (14)$$

$$x_{is}, y_{jkw}, z_s \in \{0, 1\} \qquad \forall i \in I, s \in S, j \in J, k \in K, w \in W \qquad (15)$$

The objective function (1) consists of two parts. The first item is the total earliness for all ULDs. When the due date is larger than the build-up completion time, there exists earliness and its value is equal to the remaining time, which is described as the due date minus the build-up completion time. When the due date is less than the build-up completion time, there exists tardiness. Hence, the value of earliness is 0. The second item is the unit penalty cost multiplied by the total tardiness for all ULDs. Similarly, it is a non-negative value.

The presence of Q serves to accommodate various scenarios. In situations where the flight schedule is constrained (resulting in delayed ULDs missing the flight and being retained at the terminal), the objective focuses on minimising potential tardiness, which is achieved by assigning a high value to Q. In cases of flexible flight schedules (where ULD delays are permissible), the objective revolves around maximising the aggregate remaining time to enhance operational efficiency.

Constraints (2)-(9) describe the first stage, consisting of the processes of cargo arrival at

OCC, cargo consolidation and transport, and cargo unloading at the terminal. Constraint (2) separates the two consecutive truck transports that are assigned to the same unloading bay for a period of time T_3 to ensure that each truck transport will not wait at the unloading bay of the terminal. Constraint (3) restricts that the departure time is increasing. Constraint (4) guarantees continuous truck transports. Constraint (5) states that the departure time for each cargo cannot be earlier than its arrival time plus the inspection and consolidation time. Constraint (6) states that the unloading completion time for each cargo is equal to the departure time plus transportation time and unloading time. Constraint (7) guarantees that each cargo can only be assigned to one transport. Constraint (8) ensures a valid relationship. Constraint (9) guarantees that the truck loading capacity is valid.

Constraints (10)-(14) describe the second stage, the ULD build-up process. Constraint (10) guarantees that there is exactly one ULD build-up process followed by each ULD build-up process, except the last one. Constraint (11) guarantees that each ULD build-up process has a preceding and subsequent ULD build-up process, respectively, except for the first and last processes. Constraint (12) guarantees that there is at most one ULD build-up process following the first process. Constraints (13) and (14) state the relationship of build-up completion time between ULDs and the relationship among build-up completion time, unloading completion time, and build-up processing time.

Constraints (15) are domain constraints for variables.

Remarks

The objective of our study is defined as maximising the earliness and minimising the tardiness of all ULDs, driven by both real-life considerations and interest from the industrial partner behind this problem. The primary concern of the new system is to prevent delays, as the issue of ULDs missing flights carries significant implications for the terminal operator in most cases. However, solely focusing on minimising tardiness could potentially result in equivalent solutions, especially in small-scale scenarios. In such case, it becomes necessary to consider another performance aspect to obtain improved solutions. Minimising makespan is a common approach in scheduling problems, however its application in our context is not straightforward. The challenge lies in defining the makespan due to the complex nature of our problem. ULDs are our intuitive target due to their associated due dates, yet complexity arises from the fact that each ULD comprises numerous cargoes from diverse external trucks, each with distinct arrival times. Therefore,

we have chosen to consider the remaining time, or earliness, as an additional dimension of our objectives. By incorporating this aspect, we enhance the system's robustness and resilience. Additionally, we introduce a multiplier Q to accommodate various scenarios.

Model Linearisation

The earliness and tardiness in the objective function (1) are described nonlinearly. We introduce non-negative variables E_j, T_j to represent the earliness and tardiness of ULD j. Hence, the objective is linearised and can be displayed as (16). Meanwhile, a binary variable α_j is added with constraints (17)-(23).

$$\max \quad \sum_{j>0}[E_j - QT_j] \tag{16}$$

$$D_j - c_j \leq M\alpha_j \qquad \forall j \in J \tag{17}$$

$$c_j - D_j \leq M(1-\alpha_j) \qquad \forall j \in J \tag{18}$$

$$E_j \leq D_j - c_j + M(1-\alpha_j) \qquad \forall j \in J \tag{19}$$

$$E_j \leq M\alpha_j \qquad \forall j \in J \tag{20}$$

$$T_j \geq c_j - D_j \qquad \forall j \in J \tag{21}$$

$$T_j \geq 0 \qquad \forall j \in J \tag{22}$$

$$\alpha_j \in \{0,1\} \qquad \forall j \in J \tag{23}$$

In Constraints (5) and (6), there is a bilinear item $x_{is}d_s$, which computes the departure time for cargo i if $x_{is} = 1$. Note that the upper bound of the departure time for a cargo is the latest departure time. Hence, we can introduce a new variable $dd_{is} = x_{is}d_s$ to linearise the term, and set up the upper bound as $d^u = \max_j(D_j - P_j) - T_2 - T_3$. Overall, the bilinear item $x_{is}d_s$ can be replaced with dd_{is} and Constraint (24).

$$dd_{is} \geq d_s - d^u(1 - x_{is}) \qquad \forall i \in I, s \in S \tag{24}$$

Setting of Big M

Big M is adopted in mathematical models to enforce constraints on or off. The value of the Big M usually significantly affects the computational time, thus needs to be selected carefully. On the one hand, some feasible solutions may be cut off if Big M is chosen as a small value. On the other hand, the model may become numerically tricky if Big M is

chosen as an enormous value. Consequently, we define the value of Big M in constraints (13), (17), (18), and (20) as the largest value of due date among all ULDs, that is $M = max_j D_j$.

4. The Two-stage Alternating Algorithm

The mathematical model in Section 2 can be solved by off-the-shelf MIP solvers (such as CPLEX and Gurobi) to optimality when the scale is limited. For large-scale instances, the optimal solutions may not be rapidly obtained. Considering that we are investigating a decision problem at an operational level, fast computation is genuinely essential to meet the industry needs; therefore, we present a Two-stage Alternating Algorithm below to locate high-quality solutions efficiently.

Consistent with Section 2.1 and Figure 3, the first stage can be regarded as an assignment problem, and we need to decide the assignments between cargoes and transports, as well as the departure time. The second stage is a parallel machine scheduling problem where we decide the job sequence on each machine (workstation). The Two-stage Alternating Algorithm framework is shown in Fig 4. (1) First of all, we generate an initial solution for the first stage. (2) To obtain accurate solutions rapidly, we adopt a Genetic Algorithm (GA) for the second stage. Parallel machine scheduling problems have been extensively studied in the past decades. Abundant approaches have also been proposed, including the GA that has been widely implemented. Vallada & Ruiz (2011) have successfully used a tailored GA for the parallel machine scheduling problem with setup times. This paper adopts the idea to deal with the parallel machine scheduling problem with release times. (3) Based on the solutions from the second stage, we can rapidly use the solver to obtain the optimal solutions for the first stage. Consequently, alternate iterations are conducted. Each iteration starts with a better initial solution than the previous iteration, and finally, a satisfactory solution will be obtained.

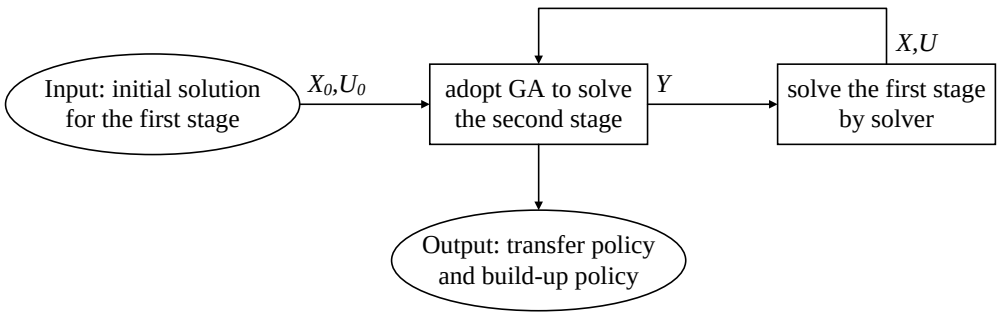


Figure 4: Alternating Algorithm framework

Hereafter, we describe the way to generate an initial solution in Section 4.1 and provide an example to interpret the local search-based crossover and explain how we solve the second stage by the tailored GA in Section 4.2. Section 4.3 summarises the alternating algorithm with pseudo-code.

4.1. Initial Solution

Each ULD has a due date and contains several cargoes, and each cargo has an arrival time. Hence, we can conclude that each ULD has a maximum remaining time (earliness), which is the due date minus all processing time and transportation time minus the latest arrival time for assigned cargoes, $D_j - P_j - T_0 - T_1 - T_2 - T_3 - \max_i a_{ij} A_i$. Therefore, we can intuitively generate the initial solution based on this rule to achieve the maximum remaining time. The detailed description is as follows, and Figure 5 shows an example of 3 ULDs and 11 cargoes.

1. Find the latest cargo for each ULD, cargo A, E, and J, respectively, and sort the arrival time for all the latest cargoes.

2. Determine the departure sequence according to the increasing order of these arrival times. Because cargo E arrives earliest compared to cargo A and cargo J, all the cargoes assigned to the same ULD as cargo E, that is, cargo D, E, F, G, will be transferred earlier than other cargoes. Cargo J arrives latest compared to cargo A and cargo E. Therefore, cargo H, I, J, K will be transferred latest. If the internal truck loading capacity (L) is 3, meaning one transport cannot transfer all cargoes assigned to ULD 2 and 3, two transports are required for these cargoes. According to the arrival time sequence, we can obtain the assignments between cargoes and transports. It is worth noting that Transport 2 only carries cargo C and E in order to decrease the cargo waiting time after the unloading process.

3. Determine the departure time according to the assignments and cargo arrival time. Meanwhile, we need to ensure there is no congestion (waiting time) at the unloading bays of the terminal. Assume there are 2 unloading bays. Hence, we can designate Transport 1 and 3 to use Bay 1 and Transport 2 and 4 to use Bay 2. Consequently, all transports' departure time can be determined. In order to decrease the cargo waiting time after the unloading process inside the terminal, Transport 1 and

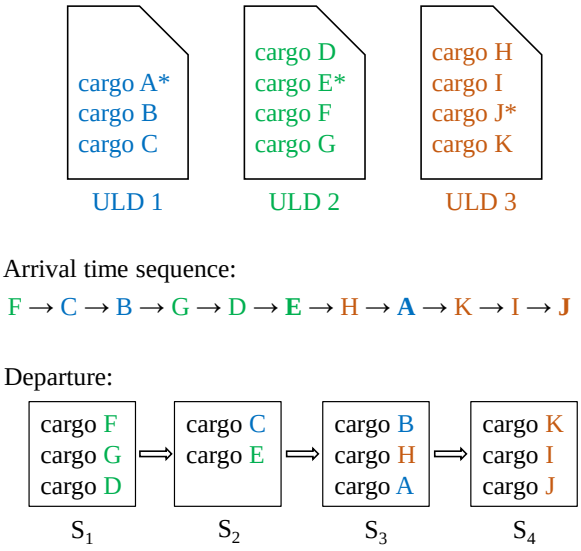


Figure 5: Example of initial solution generation

Transport 2 depart at the same time when cargo E arrives and completes processing, $d_1 = d_2 = A_E + T_0 + T_1$. Transport 3's departure time has to consider both cargo A's ready time and Transport 1's departure time to avoid any congestion, $d_3 = \max(A_A + T_0 + T_1, d_1 + T_3)$. Similarly, Transport 4's departure time $d_4 = \max(A_J + T_0 + T_1, d_1 + T_2)$.

Finally, the first stage is completed, and we can obtain the unloading completion time for the latest cargo at the terminal unloading bay, which is also the release time for each ULD build-up process.

4.2. Genetic Algorithm

A Genetic Algorithm can solve the second stage with the release time for each ULD calculated from the initial solution. GA belongs to bio-inspired optimisation approaches and has been widely adopted in machine scheduling problems. The input of GA is called population, which is a set of solutions. The existing population is regarded as parents. Crossover is conducted between two parents according to the crossover mechanism, and a new generation of individuals, called offspring, is obtained. The mutation may occur in every offspring to increase diversity. All parents and offspring will be evaluated, and the better performer will be selected as the new population. In this problem, the solution is the ULD build-up sequence on each workstation. After the crossover and mutation, a new sequence will be generated. All sequences will be compared based on the objective value. The detailed description is as follows.

1. Generate initial population. Because First-Come-First-Serve (FCFS) and First-Due-First-Serve (FDFS) are the universal policies in actual practice, we select FCFS and FDFS as two initial solutions and randomly generate the remaining solutions to fill up the population size P_{size}.

2. Local search enhanced crossover. Randomly select two parents and conduct crossover based on the crossover probability P_c. Figure 6 shows an example of crossover on two machines.

   (a) Select two parents and two crossover points, that is p_1 and p_2, on parent 1 randomly.

   (b) Cut parent 1 at the crossover points. We can get a part of offspring 1 and 2, respectively.

   (c) Compare offspring and parent 2. Offspring 1 from parent 1 already has ULD 4, 5, and 7. Delete ULD 4, 5, and 7. The remaining ULDs are the inserted ULDs.

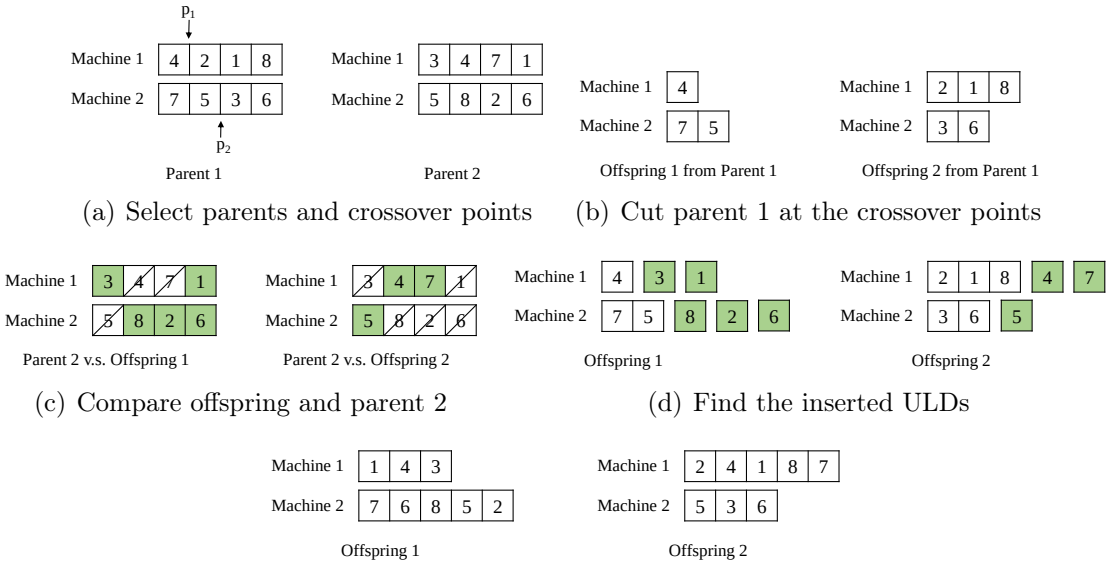
   (d) For Offspring 1 Machine 1, the inserted ULDs are ULD 3 and 1.
   For Offspring 1 Machine 2, the inserted ULDs are ULD 8, 2, and 6.
   For Offspring 2 Machine 1, the inserted ULDs are ULD 4 and 7.
   For Offspring 2 Machine 2, the inserted ULD is ULD 5.

   (e) The inserted ULDs will be placed in each position to evaluate the objective value. Finally, the best offspring can be obtained.

3. Shift mutation. A machine is randomly selected based on the mutation probability P_m. A ULD on that machine is also randomly selected and reinserted to a different position which is randomly decided.

4. Select new population. Evaluate the performance of all parents and offspring, and select the better ones to fill up the population size.

The individual with the best performance will be recorded in every iteration. An unimprovement factor of GA is introduced. When the number of times that the best performance remains unchanged reaches the value of the unimprovement factor, GA will stop and return the recorded set. The pseudo-code of the GA is shown in Algorithm 1.



(a) Select parents and crossover points (b) Cut parent 1 at the crossover points

(c) Compare offspring and parent 2 (d) Find the inserted ULDs

(e) Results of the best offspring

Figure 6: Example of crossover processes

Algorithm 1 Genetic Algorithm

Input: Population size P_{size}, crossover probability P_c, mutation probability P_m, unimprovement factor γ

Output: ULD build-up sequence Y, objective value list of each iteration of GA $GAobj$

Generate initial population including FCFS and FDFS of size P_{size}
Evaluate the objective value of each individual
Select the best individual as Y, and the best objective value as $GAobj$
unimprovement times $= 0$
while unimprovement times $< \gamma$ **do**
  for i from 1 to $\lceil \frac{P_{size}}{2} \rceil$ **do**
    Select two parents p_1 and p_2 from population randomly
    if $rand(0.0, 1.0) < P_c$ **then**
      Form the offspring c_1 and c_2 via crossover
    else
      $c_1 \leftarrow p_1$ and $c_2 \leftarrow p_2$
    end if
    if $rand(0.0, 1.0) < P_m$ **then**
      Mutate the offspring
    end if
  end for
  Evaluate the objective value of each offspring
  Form a new population of size P_{size} in descending order of the objective values of all parents and offspring
  $GAobj$.append(best objective value of the new population)
  if $GAobj[-1] > GAobj \mid GAobj[-1]$ **then**
    unimprovement times $= 0$
    $Y \leftarrow$ best individual
  else
    unimprovement times $\leftarrow$ unimprovement times $+ 1$
  end if
end while
return $GAobj$, Y

4.3. Alternating Algorithm

After the first two steps, we can get the initial solution and solve the second stage. The result of the first two steps may not be a satisfactory solution since the initial solution is generated by a greedy algorithm (although it is better than FCFS and FDFS). Besides, GA may be stuck at a local optimum. Therefore, we introduce an alternating procedure to (possibly) improve the solution quality. Based on the solution of GA, we use MIP solvers to solve the first stage rapidly. With the new solution from the first stage, we can solve the second stage by GA repeatedly. Continue with the above two alternating steps until the maximum iteration or running time. The pseudo-code of the alternating algorithm can be streamlined as Algorithm 2.

Algorithm 2 Alternating Algorithm

Input: Population size P_{size}, Crossover probability P_c, Mutation probability P_m, unimprovement factor γ

Output: Best assignment between cargoes and transports X_{best}, best ULD build-up sequence Y_{best}, objective value list of each iteration RT

Construct the initial solution of the first stage X
$X_{best} \leftarrow X$
while time limit is not reached **or** iteration limit is not reached **do**
    Calculate the release time for the second stage based on X
    Use Genetic Algorithm to solve Y based on X, obtain $GAobj$
    RT.append($max(GAobj)$)
    if $max(GAobj) > max(RT)$ **then**
        $Y_{best} \leftarrow Y$
    end if
    Solve X' of the first stage based on Y by solver, obtain objective value $Solverobj$
    $X \leftarrow X'$
    RT.append($Solverobj$)
    if $Solverobj > max(RT)$ **then**
        $X_{best} \leftarrow X$
    end if
end while
return RT, X_{best}, Y_{best}

5. Experiments and Analysis

This section evaluates the computational performance of the proposed model and solution methods. All computational experiments are implemented in Python with Gurobi 9.0.2 as the optimisation solver on a laptop with Intel(R) Core(TM) i5-8300H CPU at 2.30 GHz and 8.00 GB RAM.

As OCC is a new concept and the business data from air cargo terminal operators is confidential, the data cases and parameters are generated based on empirical advice

from frontliners in the air cargo terminal to evaluate the performance of OCC and validate the usefulness of our proposed algorithm. Each case is characterised by the number of arrival external trucks, the number of cargoes, and the number of ULDs. The arrival time is uniformly distributed between 0 and 30-minute, and each arrival external truck can carry at most 8 cargoes. Each ULD contains 5 to 10 cargoes. According to the existing requirement of the air cargo terminal operator, the external trucks need to arrive at least 2 hours before the due date of the corresponding ULD. Hence, the due dates are set to be uniformly distributed between 120 minutes and 240 minutes from the arrival time. The processing time for the ULD build-up process includes fixed time and unit time multiplied by the number of assigned cargoes. Internal truck loading capacity L is set as 8. The number of unloading bays and workstations at the terminal is set to be 3 and 2, respectively.

Based on the existing policy and universal methods used in the industry, the truck unloading process mainly adopts the first-come-first-serve policy. We maintain this policy in the first stage. For the ULD build-up process (i.e., the second stage), the most classical methods are the first-come-first-serve policy, which is to decrease the cargo waiting time before build-up processes, and the first-due-first-serve policy, which is instinctively to diminish the ULD delays. Therefore, we introduce two benchmarks: First-Come-First-Serve (FCFS) and First-Due-First-Serve (FDFS) policy. We emphasise that they both use the FCFS policy for the first stage but implement FCFS and FDFS for the second stage, respectively.

Parameter tuning experiments have been conducted on several key parameters: population size (P_{size}), crossover probability (P_c), mutation probability (P_m), and unimprovement factor (γ). These experiments were conducted using a scenario involving 6 ULDs and 30 cargoes. The parameter combination that yielded the best overall performance has been selected as the optimal setting for all subsequent numerical experiments. Comprehensive results from these parameter-tuning experiments can be found in the Appendix for further reference.

5.1. Small-scale Experiments

The small-scale cases serve as a critical validation step to demonstrate the effectiveness of our proposed method. In small-scale experiments, the number of ULDs is set to be 6, 8 and 10. Because each ULD contains 5 to 10 cargoes, the number of cargoes is set to

be the lower bound, mean value and upper bound. We compare the results of Gurobi, our proposed heuristic algorithm and two benchmarks, FCFS and FDFS. The results are shown in Table 2. The m and n in the table represent the number of ULDs and cargoes, respectively. The R in the table represents the value of the objective function in the mathematical model. The CPU represents the running time in seconds and a 20-minute time limit is given to each case. Besides, we set a 100-iteration limit for the heuristic algorithm. The running time of FCFS and FDFS for all cases is within 1 second, which is not presented in the table. The $\Delta(\%)$ is defined as the difference between the results of the proposed algorithm and the results of Gurobi or benchmarks, i.e.

$$\Delta = \frac{R_{Heuristic} - R_k}{R_{Heuristic}}, \qquad k = Gurobi, FCFS, FDFS$$

Table 2: Small-scale Cases Results

		Gurobi			Heuristic		FCFS		FDFS	
m	n	R	CPU(s)	$\Delta(\%)$	R	CPU(s)	R	$\Delta(\%)$	R	$\Delta(\%)$
6	30	694	1	0	694	8	691	0.43	691	0.43
	45	646	1	0	646	10	635	1.70	632	2.17
	60	566	1	0	566	13	551	2.65	551	2.65
8	40	1072	2	0	1072	14	1048	2.23	972	9.33
	60	772	11	-0.49	768.2	19	758	1.81	676	12.4
	80	566	8	0	566	24	518	8.48	518	8.48
10	50	1121	533	0	1121	28	1111	0.89	1081	3.57
	75	960*	1200	-0.39	956.3	33	905	5.73	838	12.7
	100	649*	1200	-0.22	647.6	44	401	38.2	499	23.1
Min.				-0.49				0.43		0.43
Mean				-0.12				6.90		8.31
Max.				0				38.2		23.1

* Feasible solution
- Solution with tardiness

Gurobi consistently delivers optimal results across all examined cases. In scenarios with relatively small quantities of cargoes and ULDs, Gurobi's efficiency is evident as it swiftly reaches the optimum outcome within mere seconds. However, as the scale increases, Gurobi encounters difficulties in finding optimal solutions within the predetermined time constraints. Specifically, in instances with 10 ULDs and 75 or 100 cargoes, Gurobi only manages to provide feasible solutions.

Recognising the inherent randomness in the GA and the alternating algorithm, we have

executed 10 sets of repeated experiments for each case utilising the heuristic algorithm. The reported results encapsulate the average values derived from these experiments. Remarkably, the proposed heuristic algorithm adeptly completes 100 iterations within one minute. In cases involving 6 ULDs, 8 ULDs and 40 cargoes, 8 ULDs and 80 cargoes, and 10 ULDs and 50 cargoes, the heuristic algorithm consistently attains the optimal value ($\Delta(\%) = 0$). In other cases, slight discrepancies are observed. The $\Delta(\%)$ of the heuristic algorithm falls within the range of (0, 0.49%), with an average value of 0.12%, marking a significant improvement compared to the two benchmark methods. Notably, one delayed ULD emerges in the scenario featuring 10 ULDs and 100 cargoes under the FCFS policy. This suggests that the benchmarks may encounter challenges in producing solutions devoid of ULD delays in larger-scale cases, which is demonstrated in Section 5.2.

5.2. Large-scale Experiments

To evaluate the performance of our proposed OCC and algorithm in the actual size, we set the number of ULDs to be 20, 22, 24, 26, 28, 30 and the number of cargoes to be the lower bound, mean value, and upper bound based on the number of ULDs. Consequently, the number of cargoes is within the range from 100 to 300. We compare the results of our proposed alternating algorithm, Gurobi, and two benchmarks, FCFS and FDFS. A 20-minute time limit is given to each case. Besides, we set a 100-iteration limit for the alternating algorithm. The unit penalty for tardiness Q is set as a standard level of 1. The results are shown in Table 3. The *delay* denotes the number of delayed ULDs in each case, which is another concern in the operation processes.

The results underscore the remarkable performance of our alternating algorithm across all scenarios. It consistently produces the highest objective values while incurring the fewest delayed ULDs in each case compared to Gurobi, FCFS and FDFS approaches. What's particularly noteworthy is that this advantage expands as the scale of the problem increases, signifying the algorithm's robust performance even in larger-scale instances.

In all cases examined, Gurobi's inability to achieve optimal solutions within the 20-minute time frame is evident. Moreover, when confronted with exceedingly large problem scales where the number of cargoes surpasses 200, Gurobi struggles to provide feasible solutions. Notably, even in such challenging conditions, our alternating algorithm manages to deliver viable solutions that outperform FCFS and FDFS.

It's acknowledged that as the numbers of cargoes and ULDs escalate, ULD delays might

occur inevitably. However, the instances of delayed ULDs stemming from our alternating algorithm are consistently fewer than those generated by Gurobi, and notably fewer in comparison to the benchmarks. This robust performance demonstrates the practical utility of our algorithm.

Table 3: Large-scale cases results

		Heuristic(average value)				Gurobi			FCFS			FDFS		
m	n	R	delay	CPU(s)	iteration	R	delay	Δ(%)	R	delay	Δ(%)	RT	delay	Δ(%)
20	100	2305	0	366	100	2299	0	0.26	2169	1	5.90	1999	0	13.28
	150	1742.1	0	721	100	1674	1	3.91	1265	5	27.39	937	0	46.21
	200	1462.9	0	1200	63	1380	2	5.67	813	8	44.43	413	2	71.77
22	110	2161	0	461	100	2161	0	0	1956	3	9.49	1567	0	27.49
	165	1308.8	0.7	936	100	1265	3	3.35	721	7	44.91	239	8	81.74
	220	1231.7	0	1200	52.3	1021	5	17.11	319	11	74.10	99	6	91.96
24	120	2749	0	571	100	2707	0	1.53	2587	4	5.89	2287	0	16.81
	180	1749.5	0.2	1200	90.9	1689	4	3.46	1013	8	42.10	831	1	52.50
	240	684.1	2.7	1200	45.1	409	11	40.21	-420	13	161.39	-888	23	229.81
26	130	1762	0	750	100	1695	3	3.8	1515	4	14.02	1325	0	24.80
	195	1918.4	0	1200	70.9	1661	4	13.42	1112	8	42.04	548	4	71.43
	260	938.4	2.9	1200	31.3	719	9	23.38	-377	13	140.17	-637	19	167.88
28	140	2518	0	981	100	2213	0	12.11	2176	6	13.58	1828	0	27.40
	210	2089.1	0.1	1200	44.5	1677	5	19.73	989	12	52.66	677	4	67.59
	280	273.6	6.7	1200	22	-117	13	142.76	-1354	19	594.88	-1699	24	720.98
30	150	2176	0	1175	100	1999	4	8.13	1462	9	32.81	1282	0	41.08
	225	2073.4	0.2	1200	39	1122	9	45.89	773	12	62.72	467	8	77.48
	300	-95.8	7.3	1200	19.8	-290	11	202.71	-1917	19	1901.04	-2667	30	2683.92

Note: Gurobi's results are the best feasible solutions found within the time limit.

5.3. Comparison with air cargo operations without OCC

To verify the practicality of OCC, we conduct the experiments under the current policy (without OCC). The comparison results are shown in Table 4. The *delay* in the table represents the number of delayed ULDs. The *longestWT* and *averageWT* mean the longest and average waiting times among all cargoes at the air cargo terminal, respectively.

Compared with the operations without OCC, the performance of our alternating algorithm still wins. As the number of ULDs and cargoes increases, the ULD delay is more frequent. However, OCC can reduce the number of delayed ULDs in all cases and even eliminate ULD delays in some cases. Besides, OCC decreases the longest and average waiting time among all cargoes at the air cargo terminal. Specifically, the longest waiting time can be substantially reduced up to 30% (214 versus 305 when $m = 20$ and $n = 200$, 350 versus 498 when $m = 30$ and $n = 300$), and the average waiting time can be extensively reduced up to 69% (33 versus 108 when $m = 20$ and $n = 150$). These results demonstrate that the benefit of OCC compared to the existing method (without OCC) could be substantial, which supports the validity of our proposal.

Table 4: Comparison with current policy

		OCC (average value)			Without OCC		
m	n	delay	longest WT	average WT	delay	longest WT	average WT
20	100	0	125	31	0	165	73
	150	0	184	33	4	243	108
	200	0	214	51	7	305	137
22	110	0	143	42	1	185	78
	165	0.7	198	41	5	254	114
	220	0	255	55	10	352	151
24	120	0	165	44	3	199	92
	180	0.2	212	57	6	279	118
	240	2.7	281	69	16	380	174
26	130	0	170	49	1	205	86
	195	0	235	61	12	326	148
	260	2.9	339	81	16	406	187
28	140	0	199	57	4	245	110
	210	0.1	253	59	10	324	141
	280	6.7	342	83	19	465	214
30	150	0	220	61	5	250	107
	225	0.2	296	74	10	341	142
	300	7.3	350	108	24	498	224

Besides the benefit of decreasing ULD delays, which greatly improves the situation of ULD missing flights, and reducing the cargo waiting time, which notably diminishes the congestion inside the air cargo terminal, we explore the cases under different truck flows between OCC and the air cargo terminal. We choose the case of 20 ULDs and 200 cargoes. The results are shown in Table 5. The $truckflow$ means the number of trucks from OCC to the air cargo terminal. Initially, without OCC, there are 60 external trucks delivering cargoes to the terminal. The instances between 40 to 50 (both included) truck flow are explored, showing that all the instances achieve a smaller longest and average waiting time compared to the current policy (without OCC). These results indicate that with the operation of OCC, the traffic flow to the airport can be decreased, which means the congestion around the airport would be recovered.

Table 5: Results of 20 ULDs and 200 cargoes under different truck flows

truck flow	40	41	42	43	44	45	46	47	48	49	50	60*
longest WT	220	207	224	218	211	220	217	210	223	234	214	305*
average WT	51	51	49	52	51	50	47	51	51	51	51	137*

* current policy

5.4. Parameter Analysis

To equitably evaluate the effects of different parameters, we conduct the parameter analysis in small-scale cases. The number of ULDs is set to be 6, 8, and 10. The number of cargoes is within the range from 30 to 100. Under these settings, Gurobi can achieve optimal results. Consequently, we can compare the results of Gurobi, FCFS, and FDFS and draw meaningful conclusions in terms of different arrival patterns, resource parameters, and time-related parameters.

1) *Arrival pattern.* The uniform distribution of previous experiments is an ideal arrival pattern. To reveal diverse situations, we investigate more arrival scenarios. Firstly, the Poisson distribution is a discrete probability distribution used for the number of events in specified intervals. Hence we adopt Poisson distribution to display the stochastic arrivals between 0 and 30-minute. Secondly, triangular distribution with the mode at 20-minute can present a situation where most truck drivers prefer to arrive late, which is very similar to peak hours. Besides, we introduce an extreme case where all trucks arrive at time 0, which indicates the backlog condition.

Twenty data sets are tested on every arrival pattern, and the box plots are shown in Fig 7. The solid line in the box plot represents the mean value, and the dashed line represents

the median. For all four arrival patterns, the performance of our model outweighs the performance of two benchmarks, FCFS and FDFS.

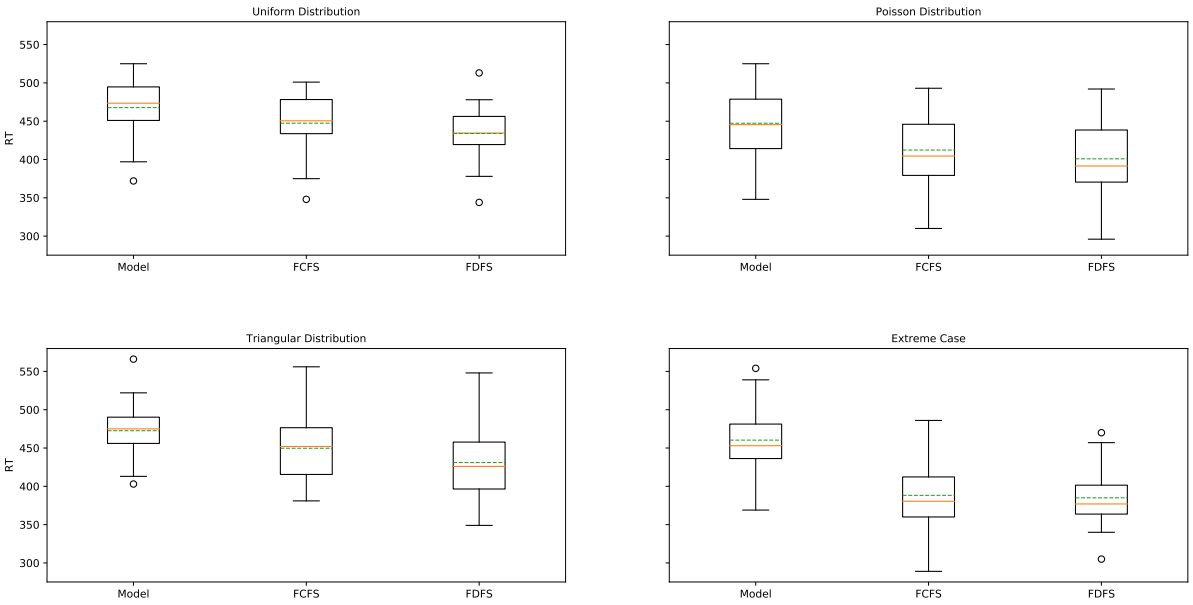


Figure 7: Box plot comparison of different arrival patterns with 20 data sets

The detailed analysis is shown in Table 6. In all cases among four arrival patterns, the improvement rates compared to the model's results are positive, demonstrating that our model has a better performance. Specifically, the improvement rates of triangular distribution are similar to that of uniform distribution and Poisson distribution, which indicates that our model is still beneficial even during peak hours. Besides, the total remaining time has an incredible improvement under the extreme case. It illustrates that the average potential improvement by introducing our model to establish coordination can be up to around 20% based on the current FCFS or FDFS policy.

Table 6: Improvement Rate of the Model's Results Compared to Benchmarks under Different Patterns

	Uniform Distribution		Poisson Distribution		Triangular Distribution		Extreme case	
	FCFS	FDFS	FCFS	FDFS	FCFS	FDFS	FCFS	FDFS
Min.	1.33%	2.34%	4.46%	4.67%	0.66%	3.32%	13.47%	16.75%
Average	4.63%	7.89%	8.67%	11.88%	5.24%	10.00%	18.90%	19.66%
Max.	9.84%	15.85%	15.45%	17.75%	11.99%	18.23%	27.68%	23.50%

2) *Resource parameters.* Resource parameters refer to the number of unloading bays B, the truck loading capacity L, and the number of workstations W. When we increase the value of these parameters, the total remaining time of all ULDs is also supposed to increase, which is shown in Figure 8. On the other hand, the total remaining time has an upper bound. When the upper bound is reached, the result will not increase even if the values of

these resource parameters keep increasing. It demonstrates the rule we proposed in Section 4.1 that each ULD has a maximum remaining time $D_j - P_j - T_0 - T_1 - T_2 - T_3 - \max_i a_{ij} A_i$, and we can use this rule to generate the initial solution for the alternating algorithm.

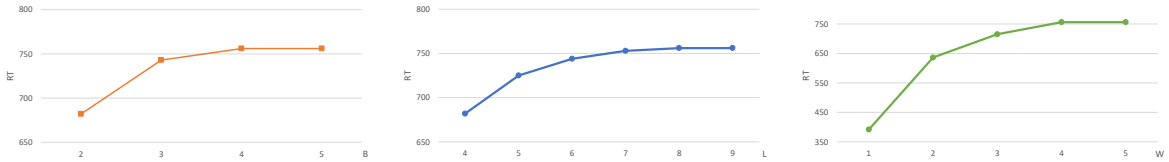


Figure 8: The results of different resource parameters

3) *Time-related parameters.* Time-related parameters refer to the processing time for cargo inspection at OCC T_0, the processing time for cargo consolidation at OCC T_1, the transportation time from OCC to terminal T_2, and the processing time for unloading a truck at the terminal T_3. The results in Figure 9 show that when the value of time-related parameters changes, the total remaining time also changes linearly. It confirms that the total remaining time has a linear relationship with time-related parameters, implying that reducing the processing time or transportation time can increase it. Besides, it reveals that the best solution (transfer policy and build-up schedule) is not affected by time-related parameters.

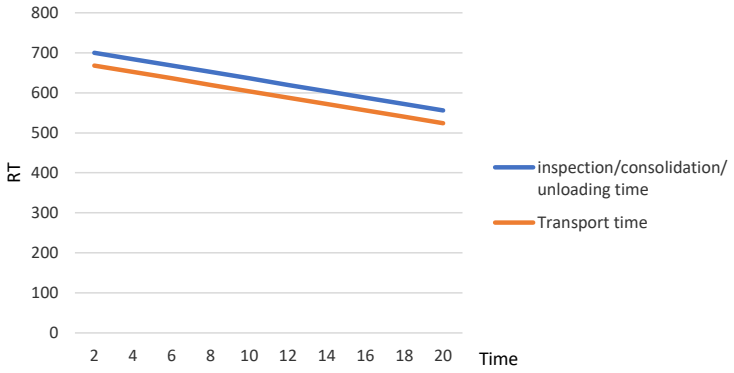


Figure 9: The results of different time-related parameters

6. Conclusions

In this paper, we investigated the two main reasons for congestion at air cargo terminals: the imbalance between the surging demand and the terminal capacity and the lack of coordination between cargo unloading and ULD build-up processes. Apart from constructing new facilities and leasing warehouses, which are the current methods adopted by some airports and companies, Offshore Consolidation Centre (OCC) is proposed to

receive arrival shipments at a distance from the airport. OCC can increase the sorting capacity to meet the growing demand and transfer the traffic pressure around the airport to alleviate the congestion. Meanwhile, OCC coordinates the cargo unloading and ULD build-up processes by simultaneously determining the transfer policy and build-up schedule to prevent ULD or flight delays.

A MILP model was formulated to describe the air cargo operation processes with OCC. A two-stage alternating algorithm embedded with a Genetic Algorithm is proposed to avoid the local optimum and obtain a satisfactory solution. Extensive numerical experiments have a similar input as actual instances and are more practical. Our alternating algorithm can obtain the largest remaining time and the least delays compared to the solver and benchmarks within a time limit.

Comparisons with air cargo operations without OCC have been conducted to justify the superiority and necessity of OCC. The results indicate that the air cargo operations with OCC have fewer delays and smaller cargo average/longest waiting time than those without OCC. Meanwhile, air cargo operators are able to decrease and control the traffic flow with the operation of OCC, which remarkably alleviates the congestion around the airport. Parameter analysis has carried out a comprehensive exploration. Different scenarios have been testified in terms of arrival patterns to accurately reveal the actual situations, such as massive late arrivals and backlog. The results demonstrate that our alternating algorithm consistently has an advantage compared to benchmarks.

In conclusion, this paper proposes a new method, Offshore Consolidation Centre, to deal with the congestion at air cargo terminals, designs an alternating algorithm for actual practice, and demonstrates its feasibility from an operational level. From an economic perspective, the feasibility of OCC is still an open problem. Besides, this paper only studies OCC's application with one air cargo terminal. Multiple terminal decisions combine VRP to this OCC operation problem, which is more complex and harder to address in the current manuscript. We would like to leave these aspects for future research.

References

Allen, J., Browne, M., Woodburn, A., & Leonardi, J. (2012). The role of urban consolidation centres in sustainable freight transport. *Transport Reviews*, *32*, 473–490.

Bartodziej, P., Derigs, U., Malcherek, D., & Vogel, U. (2009). Models and algorithms for solving combined

vehicle and crew scheduling problems with rest constraints: an application to road feeder service planning in air cargo transportation. *OR spectrum*, *31*, 405–429.

Brandt, F., & Nickel, S. (2019). The air cargo load planning problem-a consolidated problem definition and literature review on related problems. *European Journal of Operational Research*, *275*, 399–410.

Browne, M., Sweet, M., Woodburn, A., & Allen, J. (2005). Urban freight consolidation centres final report. *Transport Studies Group, University of Westminster*, *10*.

Crainic, T. G., Ricciardi, N., & Storchi, G. (2004). Advanced freight transportation systems for congested urban areas. *Transportation Research Part C: Emerging Technologies*, *12*, 119–137.

Dablanc, L., Patier, D., Gonzalez-Feliu, J., Augereau, V., Leonardi, J., Simmeone, T., Cerdà, L. et al. (2011). *SUGAR. Sustainable Urban Goods Logistics Achieved by Regional and Local Policies. City Logistics Best Practices: a Handbook for Authorities*. Technical Report.

Derigs, U., Kurowsky, R., & Vogel, U. (2011). Solving a real-world vehicle routing problem with multiple use of tractors and trailers and eu-regulations for drivers arising in air cargo road feeder services. *European Journal of Operational Research*, *213*, 309–319.

Emde, S., Abedinnia, H., Lange, A., & Glock, C. H. (2020). Scheduling personnel for the build-up of unit load devices at an air cargo terminal with limited space. *OR Spectrum*, *42*, 397–426.

Feng, B., Li, Y., & Shen, Z.-J. M. (2015). Air cargo operations: Literature review and comparison with practices. *Transportation Research Part C: Emerging Technologies*, *56*, 263–280.

Hall, R. W. (2001). Truck scheduling for ground to air connectivity. *Journal of Air Transport Management*, *7*, 331–338.

Jacyna, M. (2013). The role of the cargo consolidation center in urban logistics system. *International journal of sustainable development and planning*, *8*, 100–113.

Kasarda, J. D., & Green, J. D. (2005). Air cargo as an economic development engine: A note on opportunities and constraints. *Journal of Air Transport Management*, *11*, 459–462.

Lau, H. Y., & Zhao, Y. (2006). Joint scheduling of material handling equipment in automated air cargo terminals. *Computers in Industry*, *57*, 398–411.

Lee, C., Huang, H. C., Liu, B., & Xu, Z. (2006). Development of timed colour petri net simulation models for air cargo terminal operations. *Computers & Industrial Engineering*, *51*, 102–110.

Nordtømme, M. E., Bjerkan, K. Y., & Sund, A. B. (2015). Barriers to urban freight policy implementation: The case of urban consolidation center in oslo. *Transport Policy*, *44*, 179–186.

Ou, J., Hsu, V. N., & Li, C.-L. (2010). Scheduling truck arrivals at an air cargo terminal. *Production and Operations Management*, *19*, 83–97.

Rong, A., & Grunow, M. (2009). Shift designs for freight handling personnel at air cargo terminals. *Transportation Research Part E: Logistics and Transportation Review*, *45*, 725–739.

Selinka, G., Franz, A., & Stolletz, R. (2016). Time-dependent performance approximation of truck handling operations at an air cargo terminal. *Computers & Operations Research*, *65*, 164–173.

Vahrenkamp, R., & Berlin, L. C. (2016). 25 years city logistic: Why failed the urban consolidation centres? *European Transport-Trasporti Europei*, .

Vallada, E., & Ruiz, R. (2011). A genetic algorithm for the unrelated parallel machine scheduling problem

with sequence dependent setup times. *European Journal of Operational Research*, *211*, 612–622.

Van Heeswijk, W., Larsen, R., & Larsen, A. (2019). An urban consolidation center in the city of copenhagen: A simulation study. *International Journal of Sustainable Transportation*, *13*, 675–691.

Verlinde, S., Macharis, C., & Witlox, F. (2012). How to consolidate urban flows of goods without setting up an urban consolidation centre? *Procedia-Social and Behavioral Sciences*, *39*, 687–701.

Xu, D., Zhang, C. W., Miao, Z., & Cheung, R. K. (2014). A flow allocation strategy for routing over multiple flow classes with an application to air cargo terminals. *Computers & operations research*, *51*, 1–10.

Yan, S., Chen, C.-H., & Chen, C.-K. (2008a). Short-term shift setting and manpower supplying under stochastic demands for air cargo terminals. *Transportation*, *35*, 425–444.

Yan, S., Chen, C.-H., & Chen, M. (2008b). Stochastic models for air cargo terminal manpower supply planning in long-term operations. *Applied Stochastic Models in Business and Industry*, *24*, 261–275.

Yan, S., Chen, S.-C., & Chen, C.-H. (2006). Air cargo fleet routing and timetable setting with multiple on-time demands. *Transportation Research Part E: Logistics and Transportation Review*, *42*, 409–430.

Appendix: Parameter Tuning Experiments

Different operators lead to different performances of our algorithm. In order to fine-tune the performance of the proposed algorithm, we conducted a series of parameter-tuning experiments using a representative scenario involving 6 ULDs and 30 cargoes. The goal of these experiments was to identify the optimal values for key algorithm parameters. The parameters under consideration included the population size (P_{size}), crossover probability (P_c), mutation probability (P_m), and unimprovement factor (γ).

For each parameter, a range of values was explored to assess their impact on the algorithm's performance. The experimentation process involved systematically adjusting one parameter while keeping the others constant. This allowed us to discern how changes in each parameter affected the algorithm's convergence speed, solution quality, and stability.

The performance of the algorithm was evaluated based on the objective function value and solution convergence rate. By meticulously analysing the results, we identified the parameter combination that yielded the most promising overall performance, which was employed in all numerical experiments presented in the main text.

Population size. The population size (P_{size}) constitutes a fundamental parameter in genetic algorithms. In our investigation, we considered four distinct values for P_{size}: 50, 100, 150, and 200. The outcomes of these experiments are showcased in Figure 10. Upon inspecting the figure, an interesting trend emerges. All cases exhibit rapid initial convergence, albeit followed by a reduction in the convergence rate. It is intuitively understandable that when the search is closer to the optimal, the improvement speed is slower. The case with P_{size} at 50 displays the slowest convergence speed by iterations. Although the case with P_{size} at 200 demonstrates the fastest convergence by iterations, due to its broadest exploitation potential, it takes more time in every iteration. It's essential to consider the computational implications for larger-scale scenarios. As the problem dimensions expand, each iteration necessitates a substantial computational effort. This, in turn, could lead to significant time consumption if an iteration fails to produce improvement. Balancing these observations, we ultimately designate 100 as the default value for P_{size}. This choice optimally balances convergence speed and computational efficiency within our algorithm.

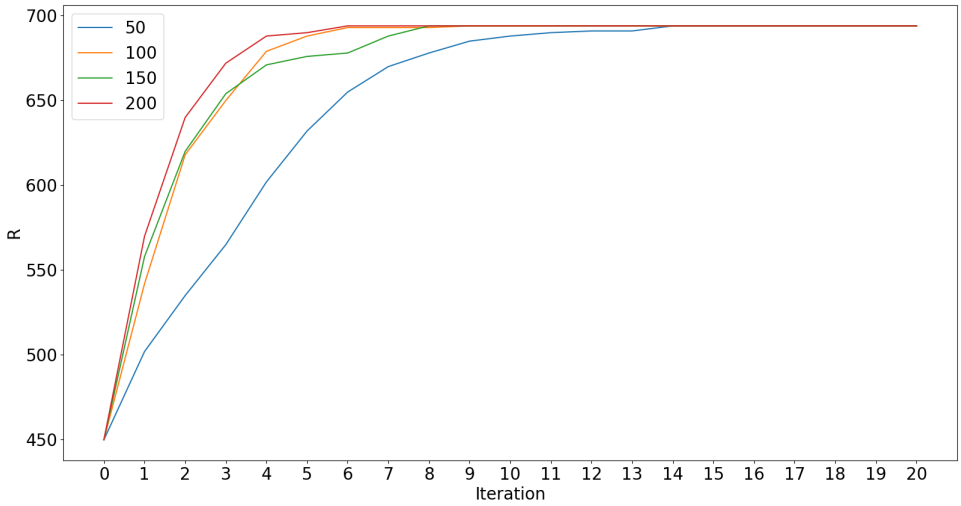


Figure 10: The objective value with different population sizes by iterations

Crossover probability. The crossover process involves selecting solutions with better performance as parents and generating new solutions that inherit their characteristics. This can be regarded as a movement towards local optima. The crossover probability (P_c) can be likened to the movement speed in this process. A higher P_c might suggest faster convergence. To delve into the insights of crossover probability, we consider values of P_c ranging from 0.1 to 1 (0.1, 0.2, 0.5, 0.8, and 1), and present the outcomes in Figure 11. The observations align with our predictions. The case with P_c of 0.1 exhibits the slowest convergence rate, while increasing P_c leads to accelerated convergence. However, it's notable that cases with P_c of 0.8 and 1 show minimal disparity. This can be attributed to the fact that all parents and offspring are merged, sorted, and selected within a single iteration. The marginal impact of accepting or rejecting a few crossover operations isn't substantial in this context. In light of these findings, we deduce that P_c has a limited effect on performance once it exceeds a small value. Therefore, we designate 0.8 as the default value for P_c. This selection optimally balances convergence speed and performance, without delving into diminishing returns associated with higher values of P_c.

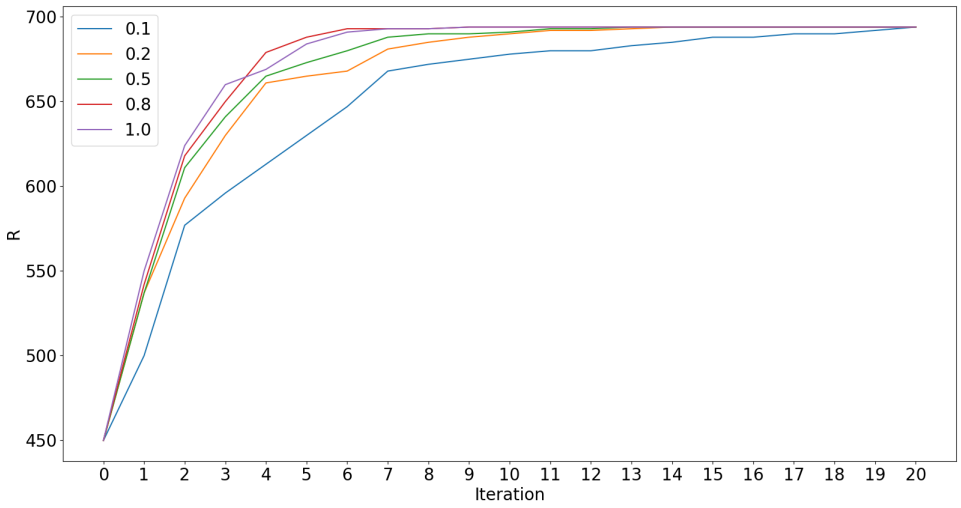


Figure 11: The objective value with different crossover probability by iterations

Mutation probability. Mutation probability (P_m) plays an important role in regulating the extent of mutation applied to all individuals collectively. In our experiments, we have considered P_m values of 0.2, 0.5, and 0.8. Additionally, we have investigated the extreme cases with P_m set to 0 and 1, representing the cases without any mutation and with mutating every offspring. The outcomes of these experiments are displayed in Figure 12. When P_m is set to 0, the algorithm becomes prone to getting trapped in local optima for certain iterations, owing to the absence of exploration beyond limited search areas. Even though cases with larger P_m indicate a more comprehensive exploration, the cases where P_m is set to 0.2, 0.5, 0.8, and 1 do not exhibit significant differences. Moreover, a larger mutation probability introduces the expense of longer computational time. Based on these insights, we designate 0.2 as the default value for P_m. to balance the potential for exploration with convergence speed.

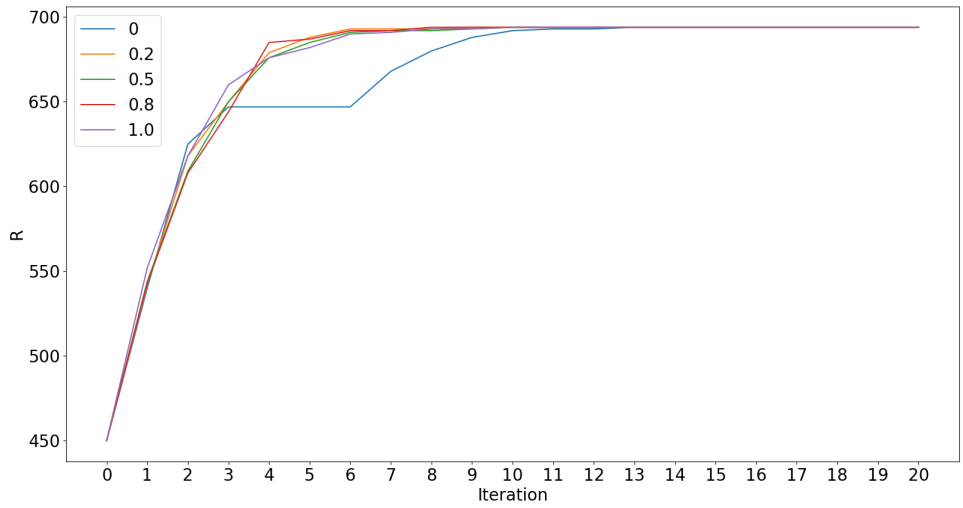


Figure 12: The objective value with different mutation probability by iterations

Unimprovement factor. The unimprovement factor (γ) serves as a termination criterion in the genetic algorithm, acting as a stop sign. When the count of consecutive unimproved iterations reaches γ, the genetic algorithm halts, and the solutions are delivered to Gurobi. In our analysis, we have considered γ values of 50, 100, 150, and 200. These results are presented in Figure 13. From our observations, it's evident that the case where γ is set to 50 exhibits the slowest convergence speed. This aligns with intuition, as a smaller γ restricts the search within the genetic algorithm. Among the three remaining cases, no substantial disparities are noticeable. Nevertheless, cases with higher γ values necessitate more time to conclude the genetic algorithm procedure. Given these insights, we designate 100 as the default value for γ to ensure sufficient searching while not excessively extending the computational time.

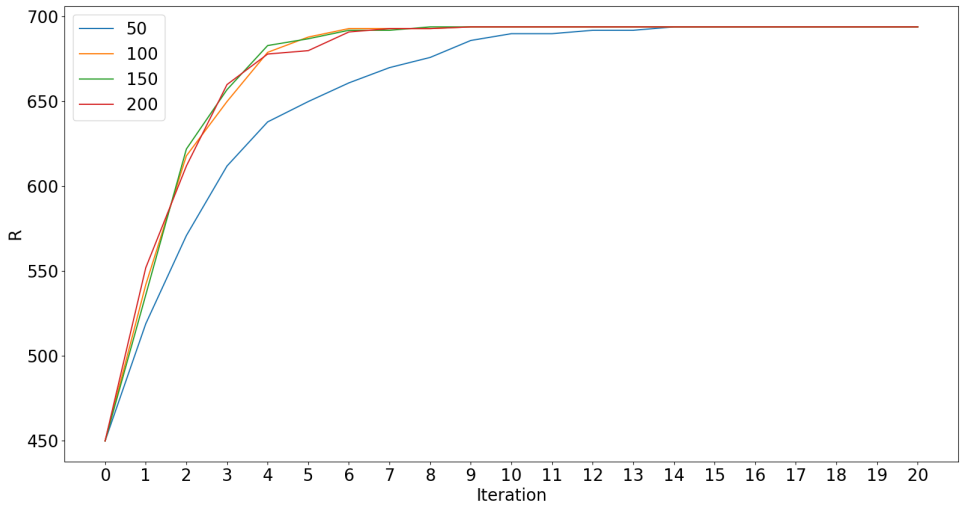


Figure 13: The objective value with different unimprovement factors by iterations