

Computers and Operations Research

Air cargo load and route planning in pickup and delivery operations

--Manuscript Draft--

Manuscript Number:	CAOR-D-22-01364
Article Type:	Research Article
Keywords:	Load Planning; Air Palletization; Weight and Balance; Pickup and Delivery; Vehicle Routing
Abstract:	<p>In aerial pickup and delivery of goods in a distribution network, transport aviation faces risks of cargo unbalancing due to the urgency required for loading, for fast take-off, and mission accomplishment, especially in times of crisis, public calamity, short deadlines for contracting customers, or any external pressure for immediate take-off. In addition, there is no technological assistance to help the load and trip planners with the huge number of demands for transport in each hub. This enables other risks, such as improper delivery, excessive fuel burn, and longer than necessary turn-around time. We defined and solved the problem of planning the loading and routing of an aircraft according to a utility score, weight and balance principles, and fuel consumption in a tour of simultaneous pickup and delivery at intermediate hubs. This NP-hard problem, named Air Cargo Load Planning with Routing, Pickup, and Delivery Problem (ACLP+RPDP), is mathematically modeled using standardized pallets in fixed positions, obeying center of gravity constraints, delivering each item to its destination, and minimizing fuel consumption costs. We performed multiple experiments with a commercial solver and well-known meta-heuristics on data based on the transport history of the Brazilian Air Force, and also designed a new method that quickly finds good solutions for a wide range of problem sizes, which solved all test scenarios within an acceptable operational runtime.</p>

Mathematical modeling of a real air transport problem that *simultaneously* involves pallet building, load balancing and route planning, in which the benefit-cost ratio regarding transport values and expenses is maximized.

Execution of tests with data from practical cases: routes with up to 7 nodes, hundreds of items to be shipped at each node, real aircraft models from *Brazilian Air Force*.

In less than one hour, this development can establish a near optimal distribution of load on pallets to be put in the cargo bay, enforcing the loaded aircraft balance, maximizing the total score and minimizing fuel consumption for the airlift.

The output, which includes the tour plan and the pallet building and arrangement plan, improves flight safety, makes ground operations more efficient, and makes sure that each item gets to its right destination.

The complete process can be executed quickly on a simple handheld computer, reducing stress for the transport planners.

Air cargo load and route planning in pickup and delivery operations

A.C.P. Mesquita, C.A.A. Sanches*

*Instituto Tecnológico de Aeronáutica - DCTA/ITA/IEC
Praça Mal. Eduardo Gomes, 50
São José dos Campos - SP - 12.228-900 - Brazil*

Abstract


In aerial pickup and delivery of goods in a distribution network, transport aviation faces risks of cargo unbalancing due to the urgency required for loading, for fast take-off, and mission accomplishment, especially in times of crisis, public calamity, short deadlines for contracting customers, or any external pressure for immediate take-off. In addition, there is no technological assistance to help the load and trip planners with the huge number of demands for transport in each hub. This enables other risks, such as improper delivery, excessive fuel burn, and longer than necessary turn-around time. We defined and solved the problem of planning the loading and routing of an aircraft according to a utility score, weight and balance principles, and fuel consumption in a tour of simultaneous pickup and delivery at intermediate hubs. This NP-hard problem, named *Air Cargo Load Planning with Routing, Pickup, and Delivery Problem (ACLP+RPDP)*, is mathematically modeled using standardized pallets in fixed positions, obeying center of gravity constraints, delivering each item to its destination, and minimizing fuel consumption costs. We performed multiple experiments with a commercial solver and well-known meta-heuristics on data based on the transport history of the *Brazilian Air Force*, and also designed a new method that quickly finds good solutions for a wide range of problem sizes, which solved all test scenarios within an acceptable operational runtime.

Keywords: Load Planning, Air Palletization, Weight and Balance, Pickup and Delivery, Vehicle Routing

1. Introduction

Air cargo transport involves several sub-problems that are difficult to solve. Recently, Brandt and Nickel [30, p. 401] defined the *Air Cargo Load Planning Problem (ACLPP)* as four sub-problems: *Aircraft Configuration Problem (ACP)*, *Build-up Scheduling Problem (BSP)*, *Air Cargo Palletization Problem (APP)* and *Weight and Balance Problem (WBP)*. Several aspects were considered in this modeling: characteristics of the items to be transported (dimensions, scores, dangerousness, etc.); types and quantities of *unit load devices (ULDs)*, commonly called pallets; when these pallets are assembled; how items are allocated to pallets; in which positions these pallets are to be placed; how the total cargo weight is balanced; etc. They also presented a comprehensive bibliographic survey of solving methods that had been developed in different situations.

However, there are still other important challenges in air cargo transport that go beyond the definition of the ACLPP, especially with regard to the flight itinerary and the loading and unloading at each destination (or node) of this travel plan. In this context, at least two more important sub-problems can be considered: pickup and delivery operations at each node, called *Pickup and Delivery Problem (PDP)*, and the search for the lowest cost route, which is the well-known *Traveling Salesman Problem (TSP)*.

Considering air cargo transport, Table 1 lists the main works in the literature in chronological order, and the corresponding sub-problems addressed. We also indicate whether the dimensions of the items were taken into account (**3D** or **2D**) and which solution method was used: heuristics (**Heu**),  (**Int**), or linear programming (**Lin**).

*Corresponding author.

Email addresses: celio@ita.br (A.C.P. Mesquita), alonso@ita.br (C.A.A. Sanches)

Table 1: Air cargo transport: literature, problems and features

Work	APP	WBP	PDP	TSP	2D	3D	Heu	Int	Lin
Larsen and Mikkelsen [1]	.	★	★	.	.
Brosh [2]	.	★	★
Ng [6]	.	★	★	.
Heidelberg <i>et al.</i> [11]	.	★	.	.	★	.	★	.	.
Mongeau and Bes [13]	★	★	★	.
Fok and Chun [14]	.	★	★	.
Chan <i>et al.</i> [16]	★	★	★	.	.
Kaluzny and Shaw [17]	.	★	.	.	★	.	.	★	.
Verstichel <i>et al.</i> [19]	.	★	★	.
Mesquita and Cunha [18]	.	.	★	.	.	.	★	.	.
Limbourg <i>et al.</i> [20]	.	★	★	.
Roesener and Barnes [21]	★	★	.	.	.	★	.	★	.
Vancroonenburg <i>et al.</i> [22]	★	★	★	.
Lurkin and Schyns [23]	.	★	★	★	.
Roesener and Barnes [24]	.	★	★	.	.
Paquay <i>et al.</i> [25, 26]	★	★	.	.	.	★	★	★	.
Chenguang <i>et al.</i> [28]	.	★	.	.	★	.	★	.	.
Wong and Ling [31]	★	★	★	.
Wong <i>et al.</i> [33]	★	★	★	.
Zhao <i>et al.</i> [34]	.	★	★	.
This article	★	★	★	★	.	.	★	★	.

As can be seen, so far Lurkin and Schyns [23] is the only work that simultaneously addresses an air cargo (WBP) and a flight itinerary (PDP) sub-problem. Although it is innovative, strong simplifications were imposed by the authors: in relation to loading, APP was ignored; with regard to routing, it is assumed a pre-defined tour plan restricted to two legs. It is important to note that these authors consider an aircraft with two doors, and the minimization of loading and unloading costs at the intermediate node was modeled through a container sequencing problem. Referring directly to this work, Brandt and Nickel [30, p. 409] comment: *However, not even these sub-problems are acceptably solved for real-world problem sizes or the models omit some practically relevant constraints.*

There are real situations that are much more complex. In this work, we consider a practical case in Brazil, which is the largest economy in Latin America. Due to its dimensions, this country has the largest air market on the continent with 2,499 registered airports, of which 1,911 are private and 588 are public. Although it is an immense distribution network, airlift missions consider 3 to 5 nodes per tour plan. Throughout this work, we address routes with up to 7 nodes, as can be seen in Table 2 and Figure 1.

Table 2: Brazilian airports distances (km)

Node IATA*	l_0 GRU	l_1 GIG	l_2 SSA	l_3 CNF	l_4 CWB	l_5 BSB	l_6 REC
GRU	0	343	1,439	504	358	866	2,114
GIG	343	0	1,218	371	677	935	1,876
SSA	1,439	1,218	0	938	1,788	1,062	676
CNF	504	371	938	0	851	606	1,613
CWB	358	677	1,788	851	0	1,084	2,462
BSB	866	935	1,062	606	1,084	0	1,658
REC	2,114	1,876	676	1,613	2,462	1,658	0

*International Air Transport Association
Source: www.airportdistancecalculator.com

**Figure 1:** A route with some Brazilian airports

In the *Brazilian Air Force* missions, hundreds of items can be carried at each node, where the objectives are to prioritise the transport of the most important items and minimize the cost of fuel along the route. As standardized pallets are used with predefined positions on the aircraft, it is possible to carry out loading and unloading at each node in around two hours. However, there is no technological assistance that guarantees the achievement of these objectives.

This work proposes a method that achieves these objectives: a pallet building and arrangement plan with a routing option that maximizes the benefit-cost ratio for the smooth execution of pickup and delivery transport missions. We develop a method that can be executed on a simple handheld computer (like a laptop or a tablet) and that provides a solution quickly enough to keep this cargo handling time under an hour. This method will reduce the stress that transport planners are subjected to, because they have to deal with a lot of information in planning the aircraft route, assembling the pallets, and picking up and delivering at each node. To the best of our knowledge, this is the first time that an air cargo transport problem that simultaneously involves APP, WBP, PDP and TSP has been addressed. This new problem is named *Air Cargo Load Planning with Routing, Pickup and Delivery Problem* (ACLP+RPDP).

This article is organized into six more sections. In Section 2, we give a brief review of the literature. In Section 3, we present the problem context and assumptions, and in Section 4, we describe the mathematical model and how we dealt with its issues. In Section 5, we describe the elaborate algorithms, whose results are presented in Section 6. Finally, our conclusions are in Section 7.

2. Related literature

In this section, we briefly describe the characteristics of the main works related to air cargo transport, following the chronological order of Table 1.

Larsen and Mikkelsen [1] developed an interactive procedure for loading 14 types of Boeing 747 into a two-leg transportation plan. Seven types of items were considered to be allocated in 17 to 42 positions. With non-linear programming and heuristics, they present a solution that minimizes positioning changes in the intermediate node, optimizing the load balancing in the aircraft.

Brosh [2] addressed the problem of planning the allocation of cargo on an aircraft. Considering volume, weight and structural constraints, the author finds the optimal load layout through a fractional programming problem.

Ng [6] developed a multi-criteria optimization approach to load the *C-130* aircraft of the *Canadian Air Force*. Based on integer programming, this model provides timely planning and improves airlift support for combat operations, solving WBP with pallets in fixed positions, and considering 20 different items.

Heidelberg *et al.* [11] developed a heuristic for 2D packing in air loading, comparing it with methods for solving the *Bin Packing Problem*. These authors conclude that the classical algorithms are inadequate in this context, because they ignore the aircraft balancing constraints.

Mongeau and Bes [13] presented a method based on linear integer programming to solve the problem of choosing and positioning containers in the *Airbus 340-300*. Safety and stability constraints were considered, with the objective of minimizing fuel consumption.

Fok and Chun [14] developed a web-based application to make efficient use of space and load balancing for an air cargo company. Based on an analysis of historical data, an operational load planning with mathematical optimization is obtained. This container load planning is usually done in roughly 2 hours before departure, when all cargo details are in place.

Chan *et al.* [16] carried out a case study with heterogeneous pallets. In order to minimize the total cost of shipping, they developed a 3D packing heuristic, with a loading plan for each pallet. Although the authors do not consider load balancing or positioning of pallets in the cargo hold, this method is relevant in commercial and industrial applications, where cargo items tend to be less dense.

Kaluzny and Shaw [17] developed a mixed integer linear programming model to arrange a set of items in a military context that optimizes the load balance.

Verstichel *et al.* [19] solved WBP by selecting the most profitable subset of containers to be loaded into an aircraft using mixed-integer programming. Experimental results on real-life data showed significant improvements compared to those obtained manually by an experienced planner.

Mesquita and Cunha [18] presented a heuristic for a real problem of the *Brazilian Air Force*, which consists of defining transport routes with simultaneous collection and delivery from a central distribution terminal.

Limbourg *et al.* [20] developed a mixed-integer program for optimally rearranging a set of pallets into a compartmentalized cargo aircraft, specifically the *Boeing 747*.

Roesener and Hall [21] solved APP and WBP as an integer programming problem, which also allows items to be loaded onto pallets according to a specific orientation (e.g., this side up).

Vancroonenbur *et al* [22] presented a mixed integer linear programming model that selects the most profitable pallets, satisfying safety and load balancing constraints on the *Boeing 747-400*. Using a solver, these authors solved real problems in less than an hour.

As already mentioned, Lurkin and Schyns [23] was the first work that simultaneously modeled WBP and PDP in air cargo transport. The authors demonstrated that this problem is NP-hard and performed some experiments with real data, noting that their model offers better results than those obtained manually.

Roesener and Barnes [24] proposed a heuristic to solve the *Dynamic Airlift Loading Problem* (DALP). Given a set of palletized cargo items that require transport between two nodes in a time frame, the objective of this problem is to select an efficient subset of aircraft, partition the pallets into aircraft loads and assign them to allowable positions in those aircraft.

Paquay *et al.* [25] presented a mathematical modeling to optimize the loading of heterogeneous 3D boxes on pallets with a truncated parallelepipeds format. Its objective is to maximize the volume used in containers, considering load balancing restrictions, the presence of fragile items and the possibility of rotating these boxes. Paquay *et al.* [26] developed some heuristics to solve this problem.

Chenguang *et al.* [28] modeled the air transport problem as a 2D packing problem, and presented a heuristic for its optimization in several aircraft, considering load balancing to minimize fuel consumption.

Wong and Ling [31] developed a mathematical model and a tool based on mixed integer programming for optimizing cargo in aircraft with different pallet configurations. Balance constraints and the presence of dangerous items were considered. Wong *et al.* [33] integrated this tool to a digital simulation model, with a visualization and validation system, based on sensors that alert about load deviations.

Zhao *et al.* [34] proposed a new modeling for WBP based on mixed integer programming. Instead of focusing on the center of gravity (CG) deviation, the authors consider the original CG envelope of the aircraft, with a linearization method for its non-linear constraints.

As can be seen, none of these works address air cargo palletization and load balancing with route optimization in a multi-leg transportation plan. This is the objective of our work: to model and elaborate a solution for a real problem that simultaneously involves 4 intractable sub-problems: APP, WBP, PDP and TSP.

3. Problem context and assumptions

In this section, we describe the context of the problem addressed in this work, as well as the assumptions considered.

3.1. Operational premises

As we are dealing with an extremely complex and diverse problem, we decided to establish some operational premises:

- Each node of the tour plan, the items to be allocated are characterized by weight, volume, scores, and previously known destinations, but the dimensions are unknown. We leave the consideration of 2D or 3D items to a future work.
- We considered a unique pallet type: the *463L Master Pallet*, a common size platform for bundling and moving air cargo. It is the primary air cargo pallet for more than 70 Air Forces and many air transport companies. This pallet has a capacity of $4500kg$ and $13.7m^3$, is equipped for locking into cargo aircraft rail systems, and includes tie-down rings to secure nets and cargo loads, which in total weighs $140kg$. For more information, see www.463LPallet.com.
- All items allocated on a pallet must have the same destination. A pallet which has not yet reached its destination may receive more items, although it is known that these operations of removing restraining nets increase handling time and the risk of improper delivery. We do not consider oversized cargo in this work, but only cargo that fit on these pallets.
- Finally, as we are interested in minimizing fuel costs, we disregarded some handling operation costs. Other costs like this may be included in future work.

Throughout this text, we call a *consolidated item* a set of items of the same destination stacked on a pallet and covered with a restraining net. It is considered unique, having the same attributes of its components, whose values are the sum of individual scores, weights and volumes. See Figure 2. To ensure accuracy in pickup and delivery operations, consolidated items must remain on board until their destination.



Figure 2: Consolidated items on 463L pallets inside a *Boeing C-17*
Source: From Wikimedia Commons, the free media repository

3.2. Aircraft and load balancing

We consider real scenarios with a smaller or a larger aircraft with payloads of 20,000kg or 75,000kg respectively. Both layouts are represented in Figures 3 and 4, where the pallets are identified by p_i .

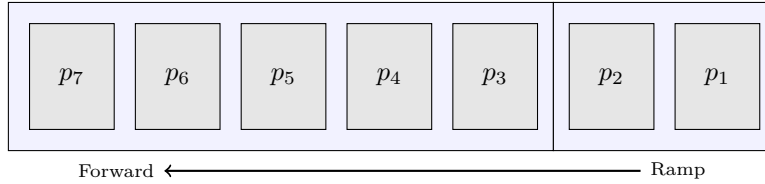


Figure 3: Smaller aircraft layout

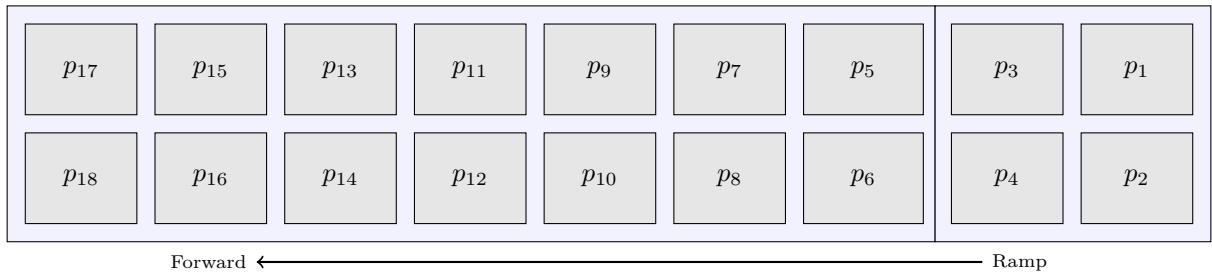


Figure 4: Larger aircraft layout

In both cases, the torque applied to the aircraft must keep its CG in the operational range, which corresponds to a percentage of the *Mean Aerodynamic Chord*¹: 0.556m in the smaller aircraft and 1.17m in the larger one. See Figure 5.

¹Chord is the distance between the leading and trailing edges of the wing, measured parallel to the normal airflow over the wing. The average length of the chord is known as the *Mean Aerodynamic Chord* (MAC).

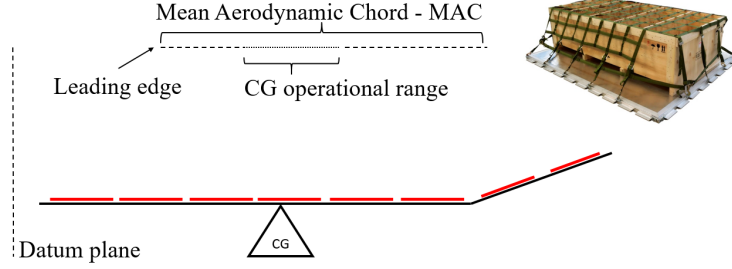


Figure 5: Aircraft longitudinal cut showing red lines as pallets

Tables 3 and 4 show the parameters in both cases. CGx and CGy refer to the relative distances of pallet centroids (in metres) in relation to the CG of aircraft along both axes. In both aircraft, as the ramps have an inclination of 25 degrees, we made the necessary corrections in CGx , $Weight$ and $Volume$ limits of the corresponding pallets. The monetary costs of both aircraft are also indicated: per unit of distance in flights legs (c_d) and per deviation in the CG (c_g). It is important to consider that c_g tends to zero as the aircraft attitude tends to level.

Table 3: Smaller aircraft parameters

Limits	Payload: 26,000kg				$limit_{long}^{CG}$: 0.556m		
Pallets	p_7	p_6	p_5	p_4	p_3	p_2	p_1
CGx (m)	-5.10	-2.70	-0.30	2.10	4.50	6.25	8.39
Weight limits (kg)	4,500	4,500	4,500	4,500	4,500	4,000	3,500
Volume limits (m^3)	13.7	13.7	13.7	13.7	13.7	8.9	6.9
Costs	c_d : US\$ 1.100/km				c_g = 0.05		

Table 4: Larger aircraft parameters

Limits	Payload: 75,000kg				$limit_{long}^{CG}$: 1.170m		$limit_{lat}^{CG}$: 0.19m		
Pallets	p_{17} p_{18}	p_{15} p_{16}	p_{13} p_{14}	p_{11} p_{12}	p_9 p_{10}	p_7 p_8	p_5 p_6	p_3 p_4	p_1 p_2
CGx (m)	-17.57	-13.17	-8.77	-4.40	0	4.40	8.77	11.47	14.89
	-17.57	-13.17	-8.77	-4.40	0	4.40	8.77	11.47	14.89
CGy (m)	1.32	1.32	1.32	1.32	1.32	1.32	1.32	1.32	1.32
	-1.32	-1.32	-1.32	-1.32	-1.32	-1.32	-1.32	-1.32	-1.32
Weight limits (kg)	4,500	4,500	4,500	4,500	4,500	4,500	4,500	4,000	3,000
Volume limits (m^3)	14.8	14.8	14.8	14.8	14.8	14.8	14.8	10.0	7.0
Costs	c_d : US\$ 4.900/km				c_g = 0.05				

We also make the following assumptions:

- on each pallet, the items are distributed in such a way that their CG coincides with the centroid of the pallet;
- the CG of the payload must be at a maximum longitudinal distance of $limit_{long}^{CG}$ from the CG of the aircraft;
- in the larger aircraft, the CG of the payload must be at a maximum lateral distance of $limit_{lat}^{CG}$ from the CG of the aircraft;
- in the larger aircraft, pallets are distributed in two identical rows (with odd and even indices, respectively), and their centroids are at a distance d_{pallet}^{CG} from the center-line of the aircraft.

3.3. Problem Summary

Informally, ACLP+RPDP can be summarised as follows:

max	(items score sum) / (tour cost) of picked up and delivered items at each node on a tour
s.t.	<p>Along a tour, the set of unvisited nodes is updated.</p> <p>In each node, an item may be included in at most one pallet.</p> <p>In each node, consolidated items are composed of items with the same destination.</p> <p>Weight, volume and score of a consolidated item are the corresponding sum of their components.</p> <p>Consolidated items remain on board until their destinations.</p> <p>Consolidated items can only be included in the same pallet if their destinations are the same.</p> <p>Only items destined to the remaining nodes can be loaded.</p> <p>The lateral and longitudinal torques must be within the operational range of the aircraft.</p> <p>Weight and volume limitations of the pallets must be respected.</p> <p>The total weight must be less than the aircraft payload or the total pallet capacity, whichever is the lowest.</p>

4. The mathematical modeling

Given the assumptions, scenarios and parameters described in the previous section, we are ready to present the mathematical modeling of ACLP+RPDP, which is one of the contributions of our work.

Let $L = \{l_0, l_1, \dots, l_K\}$ be the set of $K + 1$ nodes (or destinations), where l_0 is the origin and end of a tour. Let $d(l_i, l_j)$ be the distance from l_i to l_j , where $0 \leq i, j \leq K$. By definition, $d(l_i, l_i) = 0$. Let L_k be the set of remaining nodes when the aircraft is in l_k , $0 \leq k \leq K$. Therefore, $L_0 = L$ and $L_K = \{l_0\}$.

Let $C = \{c_{ij}\}$ be the cost matrix of flights, where $c_{ij} = c_d * d(l_i, l_j)$, $0 \leq i, j \leq K$.

Let $S_K = \{s : \{1, \dots, K\} \rightarrow \{1, \dots, K\}\}$ be the set of $K!$ permutations, which correspond to all possible tours (or itineraries) that have l_0 as origin and end, passing through the other K nodes.

Let $M = \{p_1, p_2, \dots, p_m\}$ be the set of m pallets. Each pallet p_i , $1 \leq i \leq m$, has weight capacity $p_i.w$, volume capacity $p_i.v$, pallet destinations $p_i.to[k]$, $0 \leq k \leq K$, and distance to the CG of aircraft $p_i.d$. $p_i.to[k]$ denotes that pallet p_i may assume a different destination in each node k .

Let $N_k = \{t_1^k, t_2^k, \dots, t_{n_k}^k\}$ be the set of n_k items to be loaded in node l_k , $0 \leq k \leq K$. Each item t_j^k , $1 \leq j \leq n_k$, has score $t_j^k.s$, weight $t_j^k.w$, volume $t_j^k.v$, and destination $t_j^k.to \in L_k$. Let $N = \bigcup_{0 \leq k \leq K} N_k$ be the set of items of all nodes along a tour.

Let $Q_k = \{a_1^k, a_2^k, \dots, a_{m_k}^k\}$ be the set of consolidated items loaded in $m_k \leq m$ pallets when the aircraft arrives at node l_k , with $0 \leq k \leq K$. a_i^k , $1 \leq i \leq m_k$, is the group of picked-up items that were allocated on pallet p_i in some of the previous nodes. a_i^k has total weight $a_i^k.w$, total volume $a_i^k.v$, and destination $a_i^k.to \in L_k \cup \{l_k\}$. If $a_i^k.to = l_k$, then a_i^k is unloaded, and p_i will be available for reloading; otherwise, a_i^k remains on the aircraft, eventually in another pallet, and with items of N_k having the same destination.

Let X_{ij}^k and Y_{iq}^k be binary variables, where $0 \leq k \leq K$, $1 \leq j \leq n_k$, $1 \leq i \leq m$ and $1 \leq q \leq m_k$. $X_{ij}^k = 1$ if t_j^k is assigned to p_i in node l_k , and 0 otherwise. $Y_{iq}^k = 1$ if a_q^k is assigned to p_i in node l_k , and 0 otherwise. By definition, $Y_{iq}^0 = 0$. Allocations of items or consolidated items to pallets in node l_k can be seen as a bipartite graph $G_k(V_k, E_k)$, where $V_k = M \cup N_k \cup Q_k$, $E_k = E_k^N \cup E_k^Q$, $(p_i, t_j^k) \in E_k^N$ if $X_{ij}^k = 1$, and $(p_i, a_q^k) \in E_k^Q$ if $Y_{iq}^k = 1$.

The mathematical modeling of this problem is described in the equations below.

$$\max_{\pi \in S_K} f_\pi(\tilde{s}, \tilde{c}) \quad (1)$$

$$\tilde{s} = \sum_{k=0}^K \sum_{i=1}^m \sum_{j=1}^{n_k} X_{ij}^k \times t_j^k.s \quad (2)$$

$$\tilde{c} = c_{0,\pi(1)} \times (1 + c_g \times |\epsilon_0|) + \sum_{k=1}^{K-1} [c_{\pi(k),\pi(k+1)} \times (1 + c_g \times |\epsilon_k|)] + c_{\pi(K),0} \times (1 + c_g \times |\epsilon_K|) \quad (3)$$

$$maxW = \min(Payload, \sum_{i=1}^m p_i.w) \quad (4)$$

$$\tau_k = \sum_{i=1}^m [p_i.d \times (\sum_{j=1}^{n_k} X_{ij}^k \times t_j^k.w + \sum_{q=1}^{m_k} Y_{iq}^k \times a_q^k.w)]; \quad k \in \{0, 1, \dots, K\} \quad (5)$$

$$\epsilon_k = \frac{\tau_k}{maxW \times limit_{long}^{CG}}; \quad k \in \{0, 1, \dots, K\} \quad (6)$$

$$L_0 = L; L_k = L_{k-1} - \{l_{\pi(k)}\}; \quad k \in \{1, 2, \dots, K\} \quad (7)$$

$$Y_{iq}^0 = 0; a_i^0.w = 0; a_i^0.v = 0; a_i^0.to = -1; \quad i, q \in \{1, 2, \dots, m\} \quad (8)$$

$$X_{ij}^k = 0 \text{ if } t_j^k.to \notin L_k; \quad i \in \{1, 2, \dots, m\}; \quad j \in \{1, 2, \dots, n_k\}; \quad k \in \{1, 2, \dots, K\} \quad (9)$$

$$Y_{iq}^k = 0 \text{ if } a_i^k.to \notin L_k; \quad i \in \{1, 2, \dots, m\}; \quad q \in \{1, 2, \dots, m_k\}; \quad k \in \{1, 2, \dots, K\} \quad (10)$$

$$a_i^{\pi(k+1)}.w = \sum_{j=1}^{n_k} X_{ij}^{\pi(k)} \times t_j^{\pi(k)}.w + \sum_{q=1}^{m_k} Y_{iq}^{\pi(k)} \times a_q^{\pi(k)}.w; \quad i \in \{1, 2, \dots, m_k\}; \quad k \in \{0, 1, \dots, K-1\} \quad (11)$$

$$a_i^{\pi(k+1)}.v = \sum_{j=1}^{n_k} X_{ij}^{\pi(k)} \times t_j^{\pi(k)}.v + \sum_{q=1}^{m_k} Y_{iq}^{\pi(k)} \times a_q^{\pi(k)}.v; \quad i \in \{1, 2, \dots, m_k\}; \quad k \in \{0, 1, \dots, K-1\} \quad (12)$$

$$a_i^{\pi(k+1)}.to = t_j^{\pi(k)}.to \text{ if } X_{ij}^{\pi(k)} = 1 \text{ and } t_j^{\pi(k)}.to \in L_{k+1}; \quad i \in \{1, 2, \dots, m_k\}; \quad j \in \{1, 2, \dots, n_k\}; \quad k \in \{0, 1, \dots, K-1\} \quad (13)$$

$$LatIt_k = \sum_{i=1}^m \sum_{j=1}^{n_k} (X_{ij}^k \times t_j^k.w \times (i \% 2) - X_{ij}^k \times t_j^k.w \times (i+1) \% 2); \quad k \in \{0, 1, \dots, K\} \quad (14)$$

$$LatCons_k = \sum_{i=1}^m \sum_{q=1}^{m_k} (Y_{iq}^k \times a_q^k.w \times (i \% 2) - Y_{iq}^k \times a_q^k.w \times (i+1) \% 2); \quad k \in \{0, 1, \dots, K\} \quad (15)$$

$$s.t. : d_{pallet}^{CG} \times |LatIt_k + LatCons_k| \leq \sum_{i=1}^m p_i.w \times limit_{lat}^{CG}; \quad k \in \{0, 1, \dots, K\} \quad (16)$$

$$s.t. : |\tau_k| \leq maxW \times limit_{long}^{CG}; \quad k \in \{0, 1, \dots, K\} \quad (17)$$

$$s.t. : \sum_{i=1}^m (\sum_{j=1}^{n_k} X_{ij}^k \times t_j^k.w + \sum_{q=1}^{m_k} Y_{iq}^k \times a_q^k.w) \leq maxW; \quad k \in \{0, 1, \dots, K\} \quad (18)$$

$$s.t. : \sum_{j=1}^{n_k} X_{ij}^k \times t_j^k.w + \sum_{q=1}^{m_k} Y_{iq}^k \times a_q^k.w \leq p_i.w; \quad i \in \{1, 2, \dots, m_k\}; \quad k \in \{0, 1, \dots, K\} \quad (19)$$

$$s.t. : \sum_{j=1}^{n_k} X_{ij}^k \times t_j^k.v + \sum_{q=1}^{m_k} Y_{iq}^k \times a_q^k.v \leq p_i.v; \quad i \in \{1, 2, \dots, m_k\}; \quad k \in \{0, 1, \dots, K\} \quad (20)$$

$$s.t. : \sum_{i=1}^m X_{ij}^k \leq 1; \quad j \in \{1, 2, \dots, n_k\}; \quad k \in \{0, 1, \dots, K\} \quad (21)$$

$$s.t. : Y_{iq}^k = 1 \text{ if } t_{iq}^k.to \in L_k; q \in \{1, 2, \dots, m_k\}; k \in \{0, 1, \dots, K\} \quad (22)$$

$$s.t. : p_i.to[k] = t_j^k.to \text{ if } \zeta_{ij}^k = 1; i \in \{1, 2, \dots, m\}; j \in \{1, 2, \dots, n_k\}; k \in \{1, 2, \dots, K\} \quad (23)$$

$$s.t. : p_i.to[k] = a_q^k.to \text{ if } Y_{iq}^k = 1; i \in \{1, 2, \dots, m\}; q \in \{1, 2, \dots, m_k\}; k \in \{1, 2, \dots, K\} \quad (24)$$

The objective of this problem is to find a permutation $\pi \in S_K$ that maximizes the function $f_\pi(\tilde{s}, \tilde{c})$. In this way, the tour plan will be $l_0, l_{\pi(1)}, \dots, l_{\pi(K)}, l_0$. \tilde{s} is the total score of transported items and \tilde{c} is the total cost of fuel consumption. As can be seen, \tilde{c} corresponds to the fuel consumption due to the flights carried out and the CG deviation of the transported cargo. Throughout this work, for simplicity, we use $f = \tilde{s}/\tilde{c}$.

The maximum load will be the minimum between the payload and the capacity supported by the pallets. Considering the maximum longitudinal distance allowed for the CG, all torques and deviations are calculated.

For each step of the tour, the set of unvisited nodes is updated. Although there are no items consolidated at the beginning of the tour, we defined these variables for ease of notation. Items destined outside the rest of the tour will not be loaded.

Consolidated items appear when there are items on the pallets that will not be unloaded on the next node. Weights and volumes correspond to all the items that were on the pallet, since all these items have the same destination. On subsequent nodes, consolidated items can be allocated with other items of same destination.

Equations 14 and 15 respectively, are applied only to the larger aircraft, and calculate the lateral torques of items and consolidated items loaded in both rows of pallets, whose constraint is described in 16. Similarly, 17 is the longitudinal torque constraint, which is applied to both aircraft sizes.

The weight limitation of the aircraft must be respected. The sum of weights and volumes in each pallet must not exceed its capacity. Each item is associated with a pallet at most.

Consolidated items remain on board until their destinations. At each node, an item and a consolidated item must only be allocated to a pallet if the destinations are the same.

5. Resolution strategy

Once the assumptions of this work and the mathematical modeling of the problem are presented, it is easy to see that ACLP+RPDP is NP-hard. In a similar way to Lurkin and Schyns [23, p. 6], consider the simple case where $K = 1$ (one leg), $m = 2$ (two pallets around the aircraft CG), $2n$ sufficiently light items with same scores in l_0 , and no items in l_1 . Under these conditions, through polynomial reductions for the *Set-Partition Problem*, it is possible to demonstrate that the decision problem associated with ACLP+RPDP is NP-complete.

Real cases are more complex as they have hundreds of different items in each node and involve four intractable sub-problems: APP, WBP, PDP and TSP. Through the mathematical modeling presented in the previous section, we verify that *Mixed-Integer Programming* (MIP) is not able to solve these cases in feasible time. Thus, it is necessary to adopt some strategy to find a viable solution, not necessarily optimal, that seeks to maximize the objective function f .

Our strategy is based on the fact that, in real cases, K is usually small. Specifically, we will consider $K \leq 6$ throughout this work, which is a higher value than usual in *Brazilian Air Force* missions. As a result, if we have fast node-by-node solutions that allow us to construct a complete tour, we will be able to test all possible $K!$ tours and thus select the one that provides the best value for the f function.

On the other hand, our tactic will be, at each shipping node, to predefine the destinations of the pallets at that node. In this way, we will reserve a number of pallets proportional to the volume demanded by each destination at the shipping node. We could have used another criterion, but it was observed in the experiments that volume is more constrictive in airlift. Once the destinations of the pallets are defined, we will use heuristics and a MIP solver to find the best possible node-by-node solutions. This strategy is summarized in Algorithm 1.

Algorithm 1 Main procedure

```
1: procedure ACLP + RPDP(scenario, volume)
2:   Let  $L, M, C$  be according to scenario ▷ Nodes, pallets and costs
3:    $N \leftarrow \text{ItemsGeneration}(\text{scenario}, \text{volume})$  ▷ Items available for shipment
4:   for each method ▷ method is a MIP solver or a heuristic
5:     for each  $\pi \in S_K$  ▷  $\pi$  is a permutation of the nodes
6:        $f_\pi \leftarrow \text{SolveTour}(\pi, L, M, C, N, \text{method})$  ▷ A solution to the permutation  $\pi$ 
7:        $\text{answer}[\text{scenario}, \text{volume}, \text{method}] \leftarrow \max f$  ▷ Best result among all permutations
8:   return answer
```

In this algorithm, there are six values for the *scenario* parameter, according to Table 5, which defines K , the sets of nodes, the aircraft, the pallets and the costs from Tables 3 or 4 that will be used (line 2).

The parameter *volume* is a value greater than 1, which corresponds, at each node l_k , to the ratio between the sum of the volumes of the items ($\sum_{j=1}^{n_k} t_j^k \cdot v$) and the load capacity of the pallets ($\sum_{i=1}^m p_i \cdot v$). This parameter is passed to *ItemsGeneration* (line 3), responsible for creating the items to be shipped, which will be presented in the next section (Algorithm 7).

method corresponds to a MIP solver or a heuristic that we will present in subsection 5.2. The loop of lines 5-6 goes through all permutations π , where the node-by-node resolutions are performed by *SolveTour*, whose result is stored in f_π . The best result among all $K!$ tours will be the answer for *scenario*, *volume* and *method* (line 7).

Table 5: Testing scenarios

Scenario	K	L	Aircraft
1	2	$\{l_0, l_1, l_2\}$	smaller
2	2	$\{l_0, l_1, l_2\}$	larger
3	3	$\{l_0, l_1, l_2, l_3\}$	larger
4	4	$\{l_0, l_1, l_2, l_3, l_4\}$	larger
5	5	$\{l_0, l_1, l_2, l_3, l_4, l_5\}$	larger
6	6	$\{l_0, l_1, l_2, l_3, l_4, l_5, l_6\}$	larger

Next, we will present two subsections: in the first we explain how *SolveTour* is executed, while in the second we will present the heuristic developed for node-by-node resolutions.

5.1. *SolveTour* algorithm

In addition to the set of nodes, pallets, costs and items, *SolveTour*, described in Algorithm 2, receives the parameter *method*, which corresponds to a MIP solver or a heuristic for solving the node-by-node problems, and the parameter π , which is a permutation that defines the order of visits in this tour.

As we mentioned in the previous section, all tours start and end at l_0 (lines 2-3). After initializing the score and cost values (lines 4-5), there is a loop for the $K + 1$ flights (lines 6-20). Initially we set pallets destination as -1 (line 8). When the aircraft is at node l_0 , the initial graph G_1 is empty because it has no consolidated items 11. Otherwise, the set L_k of remaining nodes is updated (line 13), and *UpdateConsolidated* (line 14) returns the set of consolidated items that have not yet reached their destination and remain on board, rearranging them on the pallets to minimize CG deviation. This allocation is stored in graph G_1 (line 15).

In the context of this work, we know that $m > K$, once the aircraft has 7 or 18 pallets and $K \leq 6$, allowing there to be at least one pallet for each node to be visited. *SetPalletsDestinations* (line 16) presets the destination of each pallet based on the volume demands of the current node, without changing the pallets destination with consolidated items.

Finally, *SolveNode* includes the edges corresponding to the items shipped at the current node, returning the graph G_2 (line 17). The score and the CG deviation of this graph are calculated (line 18) and accumulated (lines 19-20), allowing the final result of this tour (line 21).

Algorithm 2 Procedure to solve tour π with *method* (MPI solver or heuristic)

```

1: procedure SolveTour( $\pi, L, M, C, N, method$ )
2:    $\pi(0) \leftarrow 0$  ▷ The first node is the base
3:    $\pi(K + 1) \leftarrow 0$  ▷ The last node is also the base
4:    $score \leftarrow 0$ 
5:    $cost \leftarrow 0$ 
6:   for  $k \leftarrow 0$  to  $K$ 
7:     for  $i \leftarrow 1$  to  $m$ 
8:        $p_i.to[k] \leftarrow -1$  ▷ Pallet destination unset
9:     if  $k = 0$ 
10:       $L_0 \leftarrow L$ 
11:      Let  $G_1(M \cup N_0, \emptyset)$ 
12:     else
13:       $L_k \leftarrow L_k - \pi(k)$  ▷  $\pi(k)$  is the node  $k$  from tour  $\pi$ 
14:       $Q_{\pi(k)}, E_{\pi(k)}^Q, M \leftarrow UpdateConsolidated(\pi(k))$ 
15:      Let  $G_1(M \cup N_{\pi(k)} \cup Q_{\pi(k)}, E_{\pi(k)}^Q)$ 
16:       $M \leftarrow SetPalletsDestinations(\pi(k))$ 
17:       $G_2 \leftarrow SolveNode(method, \pi(k), G_1)$ 
18:       $s, \epsilon \leftarrow ScoreAndDeviation(\pi(k), G_2)$ 
19:       $score \leftarrow score + s$ 
20:       $cost \leftarrow cost + c_{\pi(k), \pi(k+1)} * (1 + c_g * |\epsilon|)$ 
21:   return  $score/cost$ 

```

UpdateConsolidated, described in Algorithm 3, finds the best allocation for the consolidated items that remain on board. Initially, the set Q is created, with the consolidated items that did not reach their destination (lines 3-7). Then *MinCGDeviation* (line 8) is run through the MIP solver with the equations 25, 26 and 27, to relocate the consolidated items on the pallets minimizing total cost and ensuring that they all remain on board, one on each pallet. As there are few variables, the MIP solver returns an allocation E_k^Q very quickly. Finally, the destination of each pallet with consolidated items is updated (lines 9-12).

Algorithm 3 Procedure to update consolidated items that remain boarded on node k

```

1: procedure UpdateConsolidated( $k$ )
2:   Let  $Q_k = \{a_1^k, a_2^k, \dots, a_{m_k}^k\}$ 
3:    $Q \leftarrow \emptyset$ 
4:   for  $q \leftarrow 1$  to  $m_k$ 
5:     if  $a_q^k.to \in L_k$ 
6:        $Q \leftarrow Q \cup \{a_q^k\}$ 
7:    $m_k \leftarrow |Q|$ 
8:    $E_k^Q \leftarrow MinCGDeviation(k, Q)$ 
9:   for  $i \leftarrow 1$  to  $m$ 
10:    for  $q \leftarrow 1$  to  $m_k$ 
11:      if  $(p_i, a_q^k) \in E_k^Q$ 
12:         $p_i.to[k] \leftarrow a_q^k.to$ 
13:   return  $Q, E_k^Q, M$ 

```

$$\text{minimize } \left| \sum_{i=1}^m \sum_{q=1}^{m_k} Y_{iq}^k \times p_i.d \times a_q^k.w \right|, \text{ where } Q = \{a_1^k, a_2^k, \dots, a_{m_k}^k\} \quad (25)$$

$$s.t. : \sum_{i=1}^m Y_{iq}^k = 1; \quad q \in \{1, 2, \dots, m_k\} \quad (26)$$

$$s.t. : \sum_{q=1}^{m_k} Y_{iq}^k \leq 1; i \in \{1, 2, \dots, m\} \quad (27)$$

SetPalletsDestinations, which set pallet destination not yet defined, is described in Algorithm 4. The vectors $vol[0..K]$ and $PalCons[0..K]$ store the total volume of items and the number of pallets with consolidated items destined for each node, respectively (lines 2-21). The destinations of each pallet are defined proportionally to the volume of items, regarding the pallets with consolidated items (lines 22-31). The node with the maximum volume demand defines the destination of any remaining pallets (lines 32-34).

Algorithm 4 Procedure to set pallets destinations based on the items to be embarked on node k

```

1: procedure SetPalletsDestinations( $k$ )
2:   for  $x \leftarrow 0$  to  $K$ 
3:      $vol[x] \leftarrow 0$ 
4:      $PalCons[x] \leftarrow 0$ 
5:    $max \leftarrow 0$ 
6:    $total \leftarrow 0$ 
7:   for  $j \leftarrow 1$  to  $n_k$ 
8:      $d \leftarrow t_j^k.to$ 
9:     if  $d \in L_k$ 
10:       $vol[d] \leftarrow vol[d] + t_j^k.v$ 
11:       $total \leftarrow total + t_j^k.v$ 
12:      if  $vol[d] > vol[max]$ 
13:         $max \leftarrow d$ 
14:   for  $q \leftarrow 1$  to  $m_k$ 
15:      $d \leftarrow a_q^k.to$ 
16:      $PalCons[d] \leftarrow PalCons[d] + 1$ 
17:     if  $d \in L_k$ 
18:        $vol[d] \leftarrow vol[d] + a_q^k.v$ 
19:        $total \leftarrow total + a_q^k.v$ 
20:       if  $vol[d] > vol[max]$ 
21:          $max \leftarrow d$ 
22:   for  $x \leftarrow 0$  to  $K$ 
23:     if  $vol[x] \neq 0$ 
24:        $quant \leftarrow \max\{1, \lfloor m \times vol[x] / total \rfloor\} - PalCons[x]$  ▷ Number of pallets needed
25:        $np \leftarrow 0$ 
26:       for  $i \leftarrow 1$  to  $m$ 
27:         if  $np = quant$ 
28:           break
29:         if  $p_i.to[k] = -1$  ▷ Pallet destination in node  $k$  was unset
30:            $p_i.to[k] \leftarrow x$ 
31:            $np \leftarrow np + 1$ 
32:   for  $i \leftarrow 1$  to  $m$ 
33:     if  $p_i.to[k] = -1$ 
34:        $p_i.to[k] \leftarrow max$ 
35:   return  $M$ 

```

ScoreAndDeviation is described in Algorithm 5, which evaluates the allocation graph generated by *SolveNode*, returning the corresponding score and CG deviation. It consists of a loop that goes through all the pallets (lines 8-17), accumulating the scores (lines 12-16) and the torques (lines 13-17) of the shipped items, allowing the final calculation of the CG deviation (lines 18-20).

Algorithm 5 Procedure to calculate the solution score and the relative torque deviation on node k

```

1: procedure ScoreAndDeviation( $k, G$ )
2:   Let  $G(V_k, E_k^N \cup E_k^Q)$ 
3:   Let  $X_{ij}^k = 1$  if  $(p_i, t_j^k) \in E_k^N, 1 \leq i \leq m, 1 \leq j \leq n_k$ 
4:   Let  $Y_{iq}^k = 1$  if  $(p_i, a_q^k) \in E_k^Q, 1 \leq i \leq m, 1 \leq q \leq m_k$ 
5:    $s \leftarrow 0$ 
6:    $\epsilon \leftarrow 0$ 
7:    $weight \leftarrow 0$ 
8:   for  $i \leftarrow 1$  to  $m$ 
9:      $weight \leftarrow weight + p_i.w$ 
10:    for  $j \leftarrow 1$  to  $n_k$ 
11:      if  $X_{ij}^k = 1$ 
12:         $s \leftarrow s + t_{ij}^k.s$ 
13:         $\epsilon \leftarrow \epsilon + t_{ij}^k.w * p_i.d$ 
14:      for  $q \leftarrow 1$  to  $m_k$ 
15:        if  $Y_{iq}^k = 1$ 
16:           $s \leftarrow s + a_{iq}^k.s$ 
17:           $\epsilon \leftarrow \epsilon + a_{iq}^k.w * p_i.d$ 
18:    $maxWeight \leftarrow \min(Payload, weight)$ 
19:    $maxTorque \leftarrow maxWeight \times limit_{long}^{CG}$ 
20:    $\epsilon \leftarrow \epsilon / maxTorque$ 
21:   return  $s, \epsilon$ 

```

5.2. Node-by-node solutions

In this subsection we present two implementations of *SolveNode*(*method*, k, G), where k is the index of the current node and G is the allocation graph of the consolidated items that remain on board at node l_k . These implementations correspond to the two possible values of the *method* parameter: *MPI* (a solver) or *Shims* (a heuristic).

5.2.1. MIP solver

To find a viable solution for ACLP+RPDP, the strategy adopted in *SolveTour* previously defines the values of some variables. Concretely, all permutations between the nodes are tested, the set of nodes to be visited is updated, the consolidated items that remain on board are reallocated to minimize the CG deviation, and the pallets destination is determined according to the volume of items available for shipment. In this way, the mathematical modeling for *SolveNode*(*MIP*, k, G) becomes simpler, which finds an allocation of available items in node l_k using previously defined values of $L_k, p_i.to[k], Y_{iq}^k$ and a_q^k .

This simplified modeling is described in the equations 28 to 40, which maximizes the score of the t_j^k items to be shipped, maintaining load balancing. The variables X_{ij}^k define the set of edges E_k^N , which will be added to the graph G .

$$\max f_s = \sum_{i=1}^m \sum_{j=1}^{n_k} X_{ij}^k \times t_j^k.s \quad (28)$$

$$max_{weight} = \min(Payload, \sum_{i=1}^m p_i.w) \quad (29)$$

$$\epsilon_k = \sum_{i=1}^m [p_i.d \times (\sum_{j=1}^{n_k} X_{ij}^k \times t_j^k.w + \sum_{q=1}^{m_k} Y_{iq}^k \times a_q^k.w)] / (maxW \times limit_{long}^{CG}) \quad (30)$$

$$X_{ij}^k = 0 \text{ if } t_j^k.to \notin L_k; i \in \{1, 2, \dots, m\}; j \in \{1, 2, \dots, n_k\} \quad (31)$$

$$LatIt_k = \sum_{i=1}^m \sum_{j=1}^{n_k} (X_{ij}^k \times t_j^k \cdot w \times (i \% 2) - X_{ij}^k \times t_j^k \cdot w \times (i + 1) \% 2) \quad (32)$$

$$LatCons_k = \sum_{i=1}^m \sum_{q=1}^{m_k} (Y_{iq}^k \times a_q^k \cdot w \times (i \% 2) - Y_{iq}^k \times a_q^k \cdot w \times (i + 1) \% 2) \quad (33)$$

$$s.t. : d_{pallet}^{CG} \times |LatIt_k + LatCons_k| \leq \sum_{i=1}^m p_i \cdot w \times limit_{lat}^{CG} \quad (34)$$

$$s.t. : |\tau_k| \leq maxW \times limit_{long}^{CG} \quad (35)$$

$$s.t. : \sum_{i=1}^m (\sum_{j=1}^{n_k} X_{ij}^k \times t_j^k \cdot w + \sum_{q=1}^{m_k} Y_{iq}^k \times a_q^k \cdot w) \leq maxW \quad (36)$$

$$s.t. : \sum_{j=1}^{n_k} X_{ij}^k \times t_j^k \cdot w + \sum_{q=1}^{m_k} Y_{iq}^k \times a_q^k \cdot w \leq p_i \cdot w; \quad i \in \{1, 2, \dots, m_k\} \quad (37)$$

$$s.t. : \sum_{j=1}^{n_k} X_{ij}^k \times t_j^k \cdot v + \sum_{q=1}^{m_k} Y_{iq}^k \times a_q^k \cdot v \leq p_i \cdot v; \quad i \in \{1, 2, \dots, m_k\} \quad (38)$$

$$s.t. : \sum_{i=1}^m X_{ij}^k \leq 1; \quad j \in \{1, 2, \dots, n_k\} \quad (39)$$

$$s.t. : p_i \cdot to[k] = t_j^k \cdot to \text{ if } X_{ij}^k = 1; \quad i \in \{1, 2, \dots, m\}; \quad j \in \{1, 2, \dots, n_k\} \quad (40)$$

5.2.2. Shims heuristic

Our goal is to find a heuristic that offers a good-quality solution for the node-by-node problem. As we are testing all $K!$ tours, its runtime is a crucial requirement. Taking this into account, we perform a series of implementations based on known meta-heuristics: *Ant Colony Optimization* (ACO) [7, 9], *Noising Method Optimization* (NMO) [8, 12, 32], *Tabu Search* [4] and *Greedy Randomized Adaptive Search Procedure* (GRASP) [5]. We also considered several ideas from the literature [10, 15, 27, 29, 32], we were careful to use the same data structures and procedures in all implementations. The decision for a simple and fast heuristic is due to the need to obtain, in less than a tour, a complete solution for ACLP+RPDP, allowing its use in an operational environment.

However, when we established the runtime limit of 0.7 seconds per node, the heuristic that presented better solutions was none of the previous ones. In this subsection, we will present a new heuristic for the node-by-node problem, called *Shims*. Like in mechanics, shims are collections of spacers to fill gaps, which may be composed of parts with different thicknesses (see Figure 6). This strategy is based on a practical observation: usually, subsets of smaller and lighter items are saved for later adjustments to the remaining clearances.

The selection of edges for E_k^N uses the *edge attractiveness* θ_{ij}^k [41], which can be understood as the tendency to allocate the item t_j^k to the pallet p_i . It is directly proportional to the score, and inversely proportional to the volume and the torque of each item.

$$\theta_{ij}^k = \frac{t_j^k \cdot s}{t_j^k \cdot v} \times (1 - \frac{t_j^k \cdot w \times |p_i \cdot d|}{\max_s \{t_s^k \cdot w\} \times \max_q \{|p_q \cdot d|\}}); \quad i \in \{1, 2, \dots, m\}, \quad j \in \{1, 2, \dots, n_k\} \quad (41)$$



Figure 6: Shims of various thicknesses
Source: www.msdirect.com/product/details/70475967

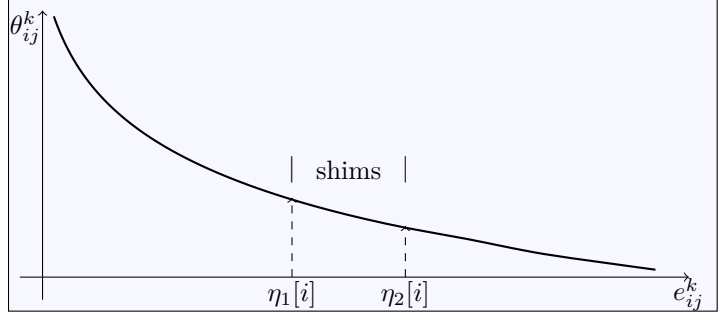


Figure 7: n_k edges e_{ij}^k of p_i sorted by θ_{ij}^k in non-ascending order

In this heuristic, described in Algorithm 6, pallets p_i are considered in non-descending order of $|p_i.d|$ (line 3). For each p_i , the n_k possible edges e_{ij}^k are considered in non-increasing order of θ_{ij}^k (line 5) in three phases: the greedy phase (lines 8-22), the composition phase of the shims (lines 23-40) and the selection phase of the best shim (lines 41-44).

Figure 7 represents the n_k possible edges e_{ij}^k of p_i sorted by θ_{ij}^k . *Shims* starts with a greedy solution, stopping at the edge with index $\eta_1[i]$ close to the local optimum (first phase). Then, considering even the edge with the index $\eta_2[i]$, it elaborates different possible complements for this pallet (second phase) and selects the best of these complements (third phase).

Algorithm 6 *Shims* heuristic on node k

```
1: procedure SolveNode(Shims,  $k$ ,  $G$ )
2:   Let  $G(V_k, E_k^Q)$ 
3:   Sort  $M$  by  $|p_i.d|$  in non-descending order
4:    $E_k^N \leftarrow \emptyset$ 
5:   Let  $E[1..m][1..n_k]$ , where  $E[i]$  is an array with  $n_k$  edges  $(p_i, t_j^k)$  sorted by  $\theta_{ij}^k$  in non-ascending order,
    $1 \leq i \leq m$ 
6:    $gap \leftarrow 2000/n_k * m$ 
7:   for  $i \leftarrow 1$  to  $m$ 
8:      $pv[i] \leftarrow 0$ 
9:      $\eta_1[i] \leftarrow 0$ 
10:    while  $(p_i.v - pv[i] > gap * p_i.v)$  and  $(\eta_1[i] < n_k)$ 
11:       $\eta_1[i] \leftarrow \eta_1[i] + 1$ 
12:       $e_{ij}^k \leftarrow E[i][\eta_1[i]]$ 
13:      if  $(e_{ij}^k$  is feasible) and  $(t_j^k.v \leq p_i.v - pv[i])$ 
14:         $E_k^N \leftarrow E_k^N \cup \{e_{ij}^k\}$ 
15:         $pv[i] \leftarrow pv[i] + t_j^k.v$ 
16:       $slack[i] \leftarrow p_i.v - pv[i]$ 
17:       $\eta_2[i] \leftarrow \eta_1[i]$ 
18:      repeat
19:         $\eta_2[i] \leftarrow \eta_2[i] + 1$ 
20:         $e_{ij}^k \leftarrow E[i][\eta_2[i]]$ 
21:         $pv[i] \leftarrow pv[i] + t_j^k.v$ 
22:      until  $(\eta_2[i] = n_k)$  or  $(pv[i] \geq (1 + 3 * gap) * p_i.v)$ 
23:       $Set \leftarrow \emptyset$ 
24:       $q \leftarrow 1$ 
25:       $shim[q] \leftarrow \emptyset$ 
26:       $volume \leftarrow 0$ 
27:      for  $x \leftarrow \eta_1[i]$  to  $\eta_2[i]$ 
28:         $e_{ij}^k \leftarrow E[i][x]$ 
29:        if  $(e_{ij}^k \notin E_k^N \cup shim[q])$  and  $(e_{ij}^k$  is feasible)
30:          if  $t_j^k.v + volume \leq slack[i]$ 
31:             $shim[q] \leftarrow shim[q] \cup \{e_{ij}^k\}$ 
32:             $volume \leftarrow volume + t_j^k.v$ 
33:          else
34:             $Set \leftarrow Set \cup shim[q]$ 
35:             $q \leftarrow q + 1$ 
36:             $shim[q] \leftarrow \emptyset$ 
37:             $volume \leftarrow 0$ 
38:            if  $t_j^k.v \leq slack[i]$ 
39:               $shim[q] \leftarrow shim[q] \cup \{e_{ij}^k\}$ 
40:               $volume \leftarrow volume + t_j^k.v$ 
41:       $sh_w \leftarrow S$ , where  $S \in Set$  and  $\sum_{e_{ab}^k \in S} t_b^k.w$  is maximum
42:       $sh_v \leftarrow S$ , where  $S \in Set$  and  $\sum_{e_{ab}^k \in S} t_b^k.v$  is maximum
43:       $sh_{best} \leftarrow S$ , where  $S \in \{sh_w, sh_v\}$  and  $\sum_{e_{ab}^k \in S} t_b^k.s$  is maximum
44:       $E_k^N \leftarrow E_k^N \cup sh_{best}$ 
45:  return  $G(V_k, E_k^N \cup E_k^Q)$ 
```

In the first phase (lines 8-22), a greedy and partial solution for p_i is constructed by edges inclusion, until a certain slack or clearance is reached. This slack was empirically defined as $gap * p_i.v$ (line 10), where $gap = 2000/n_k * m$ (line 6).

In the second phase (lines 23-40), a set of shims named *Set* is created, where each shim is formed by a group of edges in the range $[\eta_1[i], \eta_2[i]]$, whose total volume is limited by $slack[i]$. In this phase, the heuristic that provided the best results, both in terms of time and quality, is based on *Next-Fit*, which is an approximation algorithm for the *Bin Packing Problem* [3]. Basically, shims are created by accumulating the following edges, taking $slack[i]$ as a limit.

In the third phase (lines 41-44), the best shim in *Set* is chosen. Initially, two shims are found: sh_w with larger weight and sh_v with larger volume. Among the two, the shim with the highest score will be chosen and its edges are inserted into E_k^N .

6. Implementation and results

This section is composed of two parts: the generation of test instances and the results obtained in our implementation.

6.1. Instances generation

As we are dealing with a new problem, which until now had not been modeled in the literature, we have to create our own benchmarks. For this, we based on the characteristics of real airlifts carried out by the *Brazilian Air Force*, as described below.

In the military airlift carried out in Brazil from 2008 to 2010, 23% of the items weighed between 10kg and 20kg, 22% from 21kg to 40kg, 24% from 41kg to 80kg, 23% from 81kg to 200kg, and 8% between 201kg and 340kg. These five groups of items are described in Table 6, where P represents the group probability. On the other hand, the average density of these items is approximately $246kg/m^3$.

Table 6: Items weight distribution (kg) in Brazil

<i>Group</i>	<i>P</i>	<i>low</i>	<i>high</i>
1	0.23	10	20
2	0.22	21	40
3	0.24	41	80
4	0.23	81	200
5	0.08	201	340

In the generation of test instances, we use three types of random selections:

- *RandomReal*(r_1, r_2): randomly selects a real number in $[r_1, r_2]$, where r_1 and r_2 are real numbers;
- *RandomInt*(i_1, i_2): randomly selects a integer number in $[i_1, i_2]$, where i_1 and i_2 are integer numbers;
- *Roulette*(*set*) biased through ϕ : selects an element from *set*, where the probability of each element is proportional to the value of a given function ϕ defined on *set*.

ItemsGeneration, which generates N , is described in Algorithm 7. The parameter *scenario* defines L and M (line 2), and the parameter *volume* sets a limit on the total volume of items at each node (line 3). To avoid simply loading all items, we use $volume > 1$.

Algorithm 7 Procedure to generate items

```
1: procedure ItemsGeneration(scenario, volume)
2:   Let  $L$  and  $M$  be according to scenario
3:    $limit \leftarrow volume \times \sum_{i=1}^m p_i \cdot v$ 
4:   for  $k \leftarrow 0$  to  $K$ 
5:      $N_k \leftarrow \emptyset$ 
6:      $j \leftarrow 0$ 
7:      $vol \leftarrow 0$ 
8:     while  $vol < limit$ 
9:        $j \leftarrow j + 1$ 
10:      repeat
11:         $t_j^k.to \leftarrow RandomInt(0, K)$ 
12:      until  $t_j^k.to \neq k$ 
13:       $x = Roulette(item)$  biased through  $P$ 
14:       $t_j^k.w \leftarrow RandomReal(low(x), high(x))$ 
15:       $t_j^k.s \leftarrow \lfloor 100 \times (1 - \log_{10}(RandomInt(1, 9))) \rfloor$ 
16:       $t_j^k.v \leftarrow t_j^k.w / RandomReal(148, 344)$ 
17:       $vol \leftarrow vol + t_j^k.v$ 
18:       $N_k \leftarrow N_k \cup \{t_j^k\}$ 
19:       $n_k \leftarrow j$ 
20:    $N \leftarrow \bigcup_{0 \leq k \leq K} N_k$ 
21:   return  $N$ 
```

▷ From Table 6

For each generated t_j^k item, its destination is uniformly random selected (line 11), its weight has a distribution according to Table 6 (lines 13-14), its score varies 100 (highest) and 5 (lowest) according to a logarithmic scale (line 15), and its volume is randomly defined from the density, where we allow a variation of 40% more or less than the average density of $246kg/m^3$ (line 16).

6.2. Results obtained

In the tests performed, we used a 64-bit, 16GB, 3.6GHz, eight-core processor with *Linux Ubuntu 22.04.1 LTS* 64-bit as the operational system and *Python 3.10.4* as the programming language. We also used the well-known solver *Gurobi* (www.gurobi.com), version 9.5.2.

We ran Algorithm 1 in the 6 scenarios described in Table 5, considering 6 methods for node-by-node resolution: Gurobi (see 5.2.1), ACO, NMO, TS, GRASP, and Shims (Algorithm 6). In generating the items in each node, we consider 3 values for the parameter *volume*. The results obtained for the function f , with the corresponding runtime in seconds, are shown in Tables 7 ($volume = 1.2$), 8 ($volume = 1.5$) and 9 ($volume = 2.0$). For each *method*, *scenario* and *volume*, 7 different instances were generated. Therefore, 126 tests (6 scenarios \times 3 volumes \times 7 instances) were performed for each method.

The average values were presented for f and, for the runtime, the worst result obtained. To facilitate the comparison between the methods, we added a last column in these tables, where two values are indicated:

- **Normalized:** value between 0 and 1, which corresponds to the ratio between the sum of f values obtained by the method in all scenarios and the sum of the best values obtained among all methods in all scenarios. The higher the value of **Normalized**, the closer the method approached the best solutions found.
- **Speed-up:** ratio of the sums of the worst runtimes of all scenarios and the sum of the method runtimes in all scenarios. The method with the highest **Speed-up** is the fastest.

In each *scenario*, we indicate in bold the best value of f found. In each table, we also indicate in bold the best **Normalized** and **Speed-up** values.

Table 7: Methods and scenarios with $volume = 1.2$

<i>method</i>	Results	1	2	3	4	5	6	Normalized Speed-up
Gurobi	f runtime (s)	10.76 6	6.10 14	10.02 51	10.82 243	12.13 1,417	12.10 9,720	0.998 1.000
TS	f runtime (s)	8.88 4	6.56 4	8.49 17	9.19 86	11.25 516	11.48 3,600	0.900 2.709
GRASP	f runtime (s)	9.30 < 1	6.76 4	8.85 17	9.65 80	11.75 472	11.54 3,265	0.932 2.984
ACO	f runtime (s)	9.43 3	6.77 4	8.85 17	9.65 86	11.75 513	11.54 3,596	0.934 2.714
NMO	f runtime (s)	9.15 < 1	6.74 4	8.80 17	9.61 86	11.76 508	11.50 3,480	0.928 2.796
Shims	f runtime (s)	10.18 < 1	6.77 2	8.92 8	9.60 31	11.78 166	11.61 1,087	0.948 8.856

Table 8: Methods and scenarios with $volume = 1.5$

<i>method</i>	Results	1	2	3	4	5	6	Normalized Speed-up
Gurobi	f runtime (s)	15.90 6	4.756 18	8.84 68	11.94 326	16.21 1,906	16.47 12,960	0.881 1.000
TS	f runtime (s)	12.40 4	8.97 4	11.44 18	12.92 88	15.20 525	15.39 3,660	0.907 3.555
GRASP	f runtime (s)	13.12 < 1	9.36 4	12.34 17	13.63 86	16.18 515	15.44 3,564	0.951 3.651
ACO	f runtime (s)	13.17 4	9.36 5	12.34 19	13.63 90	16.18 539	15.44 3,720	0.952 3.492
NMO	f runtime (s)	12.94 2	9.36 5	12.33 19	13.60 91	16.16 544	15.43 3,780	0.948 3.442
Shims	f runtime (s)	14.15 < 1	9.39 3	12.42 11	13.72 48	16.28 265	15.49 1,720	0.968 7.470

Table 9: Methods and scenarios with $volume = 2.0$

<i>method</i>	Results	1	2	3	4	5	6	Normalized Speed-up
Gurobi	f runtime (s)	22.45 8	4.92 29	6.58 106	6.34 493	7.24 2,833	8.15 19,380	0.457 1.000
TS	f runtime (s)	18.87 4	13.72 5	17.53 19	19.28 93	21.78 556	22.35 3,900	0.929 4.992
GRASP	f runtime (s)	19.36 2	13.99 6	19.39 21	19.86 97	23.50 571	22.57 3,960	0.973 4.906
ACO	f runtime (s)	19.36 4	13.99 6	19.39 23	19.86 105	23.50 597	22.57 4,080	0.973 4.745
NMO	f runtime (s)	18.85 4	13.99 9	19.39 27	19.85 120	23.50 681	22.56 4,620	0.969 4.184
Shims	f runtime (s)	20.46 < 1	14.00 7	19.44 20	19.94 84	23.55 450	22.58 2,918	0.984 6.570

In Tables 7, 8 and 9, *Gurobi* was the slowest method and *Shims* was the fastest. This was expected, as *Gurobi* is a MIP solver, which seeks an optimal solution within the established constraints, usually reaching its time limit on each node, while *Shims* is a fast heuristic. This way, *Gurobi* even demands prohibitive times (more than 5 hours in scenario 6 of Table 9), while the maximum time required by *Shims* did not reach one hour.

Considering the **Normalized** values, in Table 7 *Gurobi* was the best (0.998), followed closely by *Shims*

(0.948). In Tables 8 and 9, *Shims* was the best (0.968 and 0.984). As expected, the performance of *Gurobi* drops sharply as the volume of shipments at the nodes increases: 0.881 in Table 8 and 0.457 in Table 9.

It is worth noting that the other heuristics always obtained **Normalized** values greater than or equal to 0.9, and were always faster than *Gurobi*.

7. Conclusions

In this work, we modeled and solved a real air transport problem, named *Air Cargo Load Planning with Routing, Pickup and Delivery Problem* (ACLP+RPDP). For the first time in the literature, an NP-hard problem that involves *simultaneously* pallet assembly, load balancing and route planning is addressed, where the cost-effectiveness of transport is maximized. We adopted some simplifications that are not critical, but that allowed the unprecedented resolution of this problem considering more than two nodes.

We consider that, in practical cases, the number K of nodes, excluding the base, is small ($K \leq 6$), each of them with hundreds of items to be shipped. Thus, considering real aircraft models, we developed some node-by-node heuristics, which allow us to enumerate all $K!$ tours and choose the best. The complete process can be executed quickly on a simple handheld computer, offering good results and reducing stress for the transport planners. As validation, we performed tests in several scenarios with real data from *Brazilian Air Force*.

In less than one hour, this development can establish a near optimal distribution of load on pallets to be put in the cargo bay, enforcing the loaded aircraft balance, maximizing the total score and minimizing fuel consumption for the airlift. The output, which includes the tour plan and the pallet building and arrangement plan, is an essential part of airlift: it improves flight safety, makes ground operations more efficient, and makes sure that each item gets to its right destination.

Our main contribution was the mathematical modeling of this complex problem, which involves four NP-hard subproblems, and a complete process that offers a fast and good-quality solution. As future work, we intend to include dimensions in the items, and consider the use of multiple aircraft or load sequencing in an aircraft with two doors.

The simple heuristic *Shims* presented the best node-by-node results in a restricted time interval. Without it, it would not be possible to test all routes: after all, the greater the number of nodes, the shorter the runtime limit for node-by-node resolution. We also show that, as the shipment volume increases, the exact node-by-node resolution becomes unfeasible, indicating the importance of a fast and good-quality heuristic.

Finally, by focusing on node-by-node resolution, the result of this work is not exclusive to aircraft and airports: it can be adapted, for example, to ships and ports, or vehicles and warehouses, or freight cars and railways. In these cases, it would be necessary to make some changes in the modeling: for example, modify the load balancing constraints, and consider the available space in vehicles or freight cars instead of pallets.

Acknowledgments

This research was partially supported by *São Paulo Research Foundation* (FAPESP, grant 2016/01860-1).

- [1] O. Larsen and G. Mikkelsen, An interactive system for the loading of cargo aircraft, *European Journal of Operational Research*, Vol. 4 (6), pp. 367-373, 1980.
- [2] I. Brosh, Optimal cargo allocation on board a plane: a sequential linear programming approach, *European Journal of Operational Research*, Vol. 8 (1), pp. 40-46, 1981.
- [3] D.S. Johnson and M.R. Garey, A 71/60 theorem for bin packing, *Journal of Complexity*, Vol. 1 (1), pp. 65-106, 1985.
- [4] F. Glover, Future paths for integer programming and links to artificial intelligence, *Computers and Operations Research*, Vol. 13, pp. 533-549, 1986.
- [5] T.A. Feo and M.G.C. Resende, A probabilistic heuristic for a computationally difficult set covering problem, *Operations Research Letters*, Vol. 8 (2), pp. 67-71, 1989.
- [6] K.Y.K. Ng, A multicriteria optimization approach to aircraft loading, *Operations Research*, Vol. 40 (6), pp. 1200-1205, 1992.

- [7] M. Dorigo, *Optimization, Learning and Natural Algorithms*, PhD Thesis, Politecnico di Milano, 1992.
- [8] I. Charon and O. Hudry, The noising method: a new method for combinatorial optimization, *Operations Research Letters*, Vol. 14 (3), pp. 133-137, 1993.
- [9] M. Dorigo, V. Maniezzo and A. Coloni, The ant system: optimization by a colony of cooperating agents, *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 26 (1), pp. 29-41, 1996.
- [10] S. Niar and A. Freville, A parallel tabu search algorithm for the 0-1 multidimensional knapsack problem, *Proceedings of 11th International Parallel Processing Symposium*, 1997.
- [11] K.R. Heidelberg, G.S. Parnell and J.E. Ames, Automated air load planning, *Naval Research Logistics*, Vol. 45 (8), pp. 751-768, 1998.
- [12] I. Charon and O. Hudry, The noising methods: a generalization of some metaheuristics, *European Journal of Operational Research*, Vol. 135 (1), pp. 86-101, 2001.
- [13] M. Mongeau and C. Bes, Optimization of aircraft container loading, *IEEE Transaction on Aerospace and Electronic Systems*, Vol. 39 (1), pp. 140-150, 2003.
- [14] K. Fok and A. Chun, Optimizing air cargo load planning and analysis, *Proceedings of The International Conference on Computing, Communications and Control Technologies*, 2004.
- [15] S. Fidanova, Ant colony optimization and multiple knapsack problem and model bias, *Numerical Analysis and Its Applications*, Springer Berlin Heidelberg, pp. 280-287, 2005.
- [16] F.T.S. Chan, R. Bhagwat, N. Kumar. M.K. Tiwari and P. Lam, Development of a decision support system for air-cargo pallets loading problem: a case study, *Expert Systems with Applications*, Vol. 31 (3), pp. 472-485, 2006.
- [17] B.L. Kaluzny and R.H.A.D. Shaw, Optimal aircraft load balancing, *International Transactions in Operational Research*, Vol. 16 (6), pp. 767-787, 2009.
- [18] A.C.P. Mesquita and C.B. Cunha, An integrated heuristic based on the Scatter Search metaheuristic for vehicle routing problems with simultaneous delivery and pickup in the context of the Brazilian Air Force, *Transportes*, Vol. 19, pp. 33-42, 2011.
- [19] J. Verstichel, W. Vancroonenburg, W. Souffriau and G.V. Berghe, A mixed integer programming approach to the aircraft weight and balance problem, *Procedia Social and Behavioral Sciences*, Vol. 20, pp. 1051-1059, 2011.
- [20] S. Limbourg, M. Schyns and G. Laporte, Automatic aircraft cargo load planning, *Journal of the Operational Research Society*, Vol. 63 (9), pp. 1271-1283, 2012.
- [21] A.G. Roesener and S. Hall, A nonlinear integer programming formulation for the airlift loading problem with insufficient aircraft, *Journal of Nonlinear Analysis and Optimization: Theory and Applications*, Vol. 5 (1), pp. 125-141, 2014.
- [22] W. Vancroonenburg, J. Verstichel, K. Tavernier and G.V. Berghe, Automatic air cargo selection and weight balancing: a mixed integer programming approach, *Transportation Research Part E: Logistics and Transportation Review*, Vol. 65, pp. 70-83, 2014.
- [23] V. Lurkin and M. Schyns, The airline container loading problem with pickup and delivery, *European Journal of Operational Research*, Vol. 244 (3), pp. 955-965, 2015.
- [24] A.G. Roesener and J.W. Barnes, An advanced tabu search approach to the dynamic airlift loading problem, *Logistics Research*, Vol. 9 (1), pp. 1-18, 2016.
- [25] C. Paquay, M. Schyns and S. Limbourg, A mixed integer programming formulation for the three-dimensional bin packing problem deriving from an air cargo application, *International Transactions in Operational Research*, Vol. 23 (1-2), pp. 187-213, 2016.

- [26] C. Paquay, S. Limbourg, M. Schyns and J.F. Oliveira, MIP-based constructive heuristics for the three-dimensional bin packing problem with transportation constraints, *International Journal of Production Research*, Vol. 56 (4), pp. 1581-1592, 2018.
- [27] S. Laabadi, M. Naimi, H. El Amri and B. Achchab, The 0/1 multidimensional knapsack problem and its variants: a survey of practical models and heuristic approaches, *American Journal of Operations Research*, Vol. 8, pp. 395-439, 2018.
- [28] Y. Chenguang, L. Hu and G. Yuan, Load planning of transport aircraft based on hybrid genetic algorithm, *MATEC Web of Conferences*, Vol. 179, pp. 1-6, 2018.
- [29] M.T. Alonso, R. Alvarez-Valdes and F. Parreno, A GRASP algorithm for multi-container loading problems with practical constraints, *A Quarterly Journal of Operations Research*, Vol. 18, pp. 49-72, 2019.
- [30] F. Brandt and S. Nickel, The air cargo load planning problem - a consolidated problem definition and literature review on related problems, *European Journal of Operational Research*, Vol. 275 (2), pp. 399-410, 2019.
- [31] E.Y.C. Wong and K.K.T. Ling, A mixed integer programming approach to air cargo load planning with multiple aircraft configurations and dangerous goods, *Proceedings of 7th International Conference on Frontiers of Industrial Engineering*, 2020.
- [32] S. Zhan, L. Wang, Z. Zhang and Y. Zhong, Noising methods with hybrid greedy repair operator for 0-1 knapsack problem, *Memetic Computing*, Vol. 12, pp. 37-50, 2020.
- [33] E.Y.C. Wong, D.Y. Mo and S. So, Closed-loop digital twin system for air cargo load planning operations, *International Journal of Computer Integrated Manufacturing*, Vol. 34 (7-8), pp. 801-813, 2021.
- [34] X. Zhao, Y. Yuan, Y. Dong and R. Zhao, Optimization approach to the aircraft weight and balance problem with the centre of gravity envelope constraints, *IET Intelligent Transport Systems*, Vol. 15 (10), pp. 1269-1286, 2021.