Optimization of Aircraft Container Loading

Marcel Mongeau, Christian Bès

Abstract

We address the problem of loading as much freight as possible in an aircraft while balancing the load in order to minimize fuel consumption and to satisfy stability/safety requirements. Our formulation methodology permits to solve the problem on a PC, within ten minutes, by off-the-shelf integer linear programming software. This method decides which containers to load (and in which compartment) and which to leave on the ground.

I. Introduction

IN order to make air freight more profitable, one is interested in maximizing the mass of goods loaded. However, the arrangement of the containers in the cargo holds affects the position of the center of gravity of the aircraft, which in turn has an impact on aircraft drag. One therefore wants to balance the load so that the aircraft will fly more safely, fly faster and use less fuel. We are concerned in this paper with the problem of optimizing the layout of containers within the cargo holds so as to take into account these conflicting objectives. To give an idea of the relevance of the problem: a displacement of the center of gravity of less than 75 cm in a long-range aircraft yields, over a 10 000 km flight, a saving of 4 000 kg of fuel. (Clearly, when a displacement of the center of gravity can also be achieved via an automatic fuel transfer system—available, for instance, on the Airbus family—the approach described in the current paper could still prove useful to allow more flexibility and/or to improve results.) Note also that, as it is observed in [1], another application of the balanced loading problem is the loading of trucks. Indeed, in order to minimize the maximum axle weight, one equivalently attempts to load a truck so that the center of gravity is as close as possible to the mid-point between two axles.

Attention has been focused during the last two decades on automation of the load planning process in order to expedite the plane's departure and/or to account for last minute change of the list of containers to be loaded. Airliners currently proceed heuristically through experienced ground personnel trying to obtain an acceptable loading (i.e. satisfying the above stability and structural constraints), by manual trial and error process, possibly with the aid of computer assistance, at best with interactive computerized graphics.

L. A. Martin-Vega wrote a complete review [2] of the manual and the computer-assisted approaches to the aircraft load planning problem developed before 1985. He observed that the emphasis has been on computer *assistance*, rather than automation of the load planning. The main objective of the planners were also to generate a *feasible* plan rather than an optimal one. Center of gravity considerations are generally dealt with via *pyramid loading* (assign successively the heaviest items to the central positions, alternately working towards the front and back of the aircraft). Further, Martin-Vega notes that computer assistants such as CARLO [3], AALPS [4] (or, more recently, [5]), and DMES [6] were exceptions in what he considered a manually dominated process. (To summarize, CARLO, for the Boeing 747 Combi, and AALPS, for the C-130, C-141B, and C-5A aircrafts, are interactive, heuristic, computerized procedures, as is DMES, for the C-130, C-141B, and C-5A, which moreover incorporates graphics and mini/micro computer technology). The unique mathematical programming approach of practical relevance reported by Martin-Vega is that of I. Brosh [7]. His optimization formulation of the problem is different from the one we propose in the current paper: Brosh assumes having L types

This research was prompted by Airbus France, which provided the data and financially supported students working on the project, in collaboration with the Weight Group directed by Bernard Huber. We thank the students Paul Chazottes, Caroline Combelles, Fabrice Gouazé, Samuel Grossiord, Karine Jagu, and Estelle Saez for their implementation work at Airbus France.
M. Mongeau is with the Labo MIP, Université Paul Sabatier, Toulouse, France. E-mail: mongeau@cict.fr
C. Bès, formerly with Airbus France, Toulouse, is now with the Dept Génie Mécanique, Université Paul Sabatier, Toulouse, France. E-mail: cbes@cict.fr

of *homogeneous* cargo (such as mail, bulk etc.) yielding a *continuous* optimization problem (whereas we are dealing here with the *combinatorial* optimization problem of determining which containers are to be loaded, and at which of the pre-determined locations in the holds). This easier continuous optimization framework allows Brosh to incorporate finer modeling of the linear constraints specifying feasible positions of the center of gravity in terms the aircraft gross weight. These linear constraints involving the positions of the center of gravity yields nonlinear constraints in the formulation. By contrast, we are content with simple upper- and lower-bound constraints on the positions of the center of gravity (this is acceptable for the application we consider—the long-range aircraft Airbus A340-300), which gives linear constraints in our formulation.

We now describe previous work which has been reported in the literature since 1985. A simple greedy heuristic (with an error bound on the deviation from a target position of the center of gravity) is proposed in [1] for the following special case. First, all given containers must be loaded, and secondly, the containers are to be positioned on a one-dimensional hold. Further related work was reported in [8] for a military application. The problem considered there was to airlift cargo, which again must be entirely loaded, in a specified prioritized sequence (via a goal-programming formulation solved by branch and bound, and the method relied on "taking into account the expertise of highly trained loadmasters"). More recently, another simple greedy heuristic was proposed [9] for the same special case as that considered in [1], with the difference that the algorithm in [9] has a better worst-case performance than that in [1]. Finally, in [10], Thomas et al. proposed an integer programming formulation of the problem. They addressed in fact a difficult variant of the aircraft loading problem: they are specifically concerned with the loading of lighter aircrafts. Such aircrafts are more fuel efficient, but at a cost of tighter structural and weight limits: shear limits—shear being a measure of the downward forces exerted on the plane—are in this case extremely sensitive to the loading, a problem not experienced in the heavier aircrafts. As a consequence, there is no *static* limits on the weight to be placed in each cargo zone: the zone limits on the weight are a stepwise-linear function of the center of gravity. This yields nonlinear constraints on the position of the center of gravity. This is in contrast with our context of application (A340-300, one of Airbus' long-range commercial carriers). In order to avoid an integer *nonlinear* programming problem, they assume, in a phase-1 subproblem, that the list of containers to be loaded is, again, known a priori. This phase-1 subproblem must then repeatedly be solved after removing containers from the list until a feasible packing is found. The method they presented is a heuristic. They use a spreadsheet interface in order to find a solution which satisfies the constraints (rather than maximizing the weight loaded). When this is not possible in phase 1, then the ground crew must select one or more containers to remove from the set (and repeat phase 1). Preferred-positions constraints are then added in phase 2. Such constraints are recursively removed until a feasible load plan is found.

Related problems dealing with containers to be loaded in ships or trains are addressed in the literature. Recent instances include [11] which describes post-processing heuristic approaches to produce good loading arrangements with an *even weight distribution* of the cargo. In [12], the authors discuss models and algorithms to facilitate rapid transfers of the containers between trains. Finally, [13] deals with expert systems to expedite the unloading and loading of container vessels in the Port of Singapore.

The purpose of the current paper is to introduce a new method for addressing aircraft container loading. We shall see that the mathematical programming formulation we propose can be solved to optimality (our method is not a heuristic), in a time which is acceptable for practical applications. The advantages of the method are twofold. Firstly, as opposed to previous work [1], [8], [9] presented in the literature, our method does not rely on the assumption that the containers to be loaded are known a priori (our method decides which containers are to be loaded and which are to be left on the ground for a subsequent flight). Secondly, we have demonstrated that the formulation we propose can be solved using real data in a reasonable amount of time: it can be solved by off-the-shelf integer linear programming software, it involves little memory, and can therefore be run on a PC within ten minutes.

These were specific requests from Airbus France, which wanted to propose an accompanying container loading software to its customers. The specific problem we are addressing in the current paper is as follows. Given a list of containers, with their respective weight and volume, we must assign (a subset of) the containers to a finite number of possible container locations (also given, for the specific aircraft under consideration) in the cargo holds, so that the two following (often contradictory) objectives are optimized. First, as much weight as possible should be loaded (for airliners, freight income is generally related to the *weight* loaded, as opposed to volume for example). Secondly, the resulting center of gravity of the aircraft should be as far aft as possible (in order to minimize fuel consumption), but not behind a limit imposed by stability requirements. Structural constraints include compartment-volume and compartment-weight capacities, and total aircraft maximal weight once loaded. The allowable total weight depends on the weight of the empty aircraft plus passengers, fuel, and bulk freight. Various constraints of practical relevance can easily be integrated within our mathematical formulation. For instance, a given subset of the container list can be specified to be loaded (containers that cannot be left on the ground because they contain e.g. perishable goods). Furthermore, some given containers can be constrained to be placed in some specified compartments (for instance near a door in order to be landed at a stopover, or for toxic material containers to be away from foodstuff containers, etc.) Other constraints that can easily be modeled include requiring some given containers, already loaded by the ground crew, not to be displaced, in order to expedite the plane's departure (as in the Federal Express application [10] where time is a critical factor).

The paper is organized as follows. We introduce an integer linear programming formulation of the aircraft container loading problem in the next section. Section III specifically details how the volume capacity constraints can be expressed so as to fit within the integer linear programming formulation. The formulation of these constraints are exemplified on a specific aircraft, namely the A340-300, one of Airbus' long-range commercial carriers. The overall optimization formulation is summarized in Section IV. In order to illustrate our method, we report computational experiments with typical data in Section V. We conclude in Section VI.

II. Model Formulation

We are concerned in this paper with positioning the center of gravity only along the *longitudinal* (fore and aft) axis of the aircraft, as the problem of balancing the load from side to side is commonly considered of marginal importance. The holds of the aircraft are divided into compartments (see e.g. Figure 1). In accordance with international regulations, when it comes to computing the location of the center of gravity, we proceed as if every container of a given compartment were positioned at the geometric center of that compartment. We aim at allocating each given container to one compartment (or to the ground, when not loaded). We next set the notation.

A. Given data (input)

N_{cont}	number of containers on the ground
N_{comp}	number of compartments
N_{hold}	number of holds
H_k	$(\subseteq \{1, 2, \ldots, N_{comp}\})$ are the compartments in hold k $(k = 1, 2, \ldots, N_{hold})$
M_a	mass of the aircraft (before loading)
M_i	mass of container i $(i = 1, 2, \ldots, N_{cont})$
$M_{\max}$	maximal mass of freight that can be loaded
$M^k_{\max}$	maximal mass of freight that can be loaded in hold k $(k = 1, 2, \ldots, N_{hold})$
X_a	(longitudinal) position of the center of gravity of the aircraft before loading
X_j	(longitudinal) position of the geometric center of compartment j $(j = 1, 2, \ldots, N_{comp})$
X_{target}	ideal (longitudinal) position of the center of gravity of the aircraft after loading
X_{stab}	maximal (longitudinal) position of the center of gravity of the aircraft after loading in order to satisfy stability requirements,

where

X_{target} and X_{stab} (which depends on the aircraft performance and safety requirements respectively) are given by a load-and-balance software provided by the aircraft manufacturer. Further data given as input include the following:

- The dimensions and weight of each of the N_{cont} given containers;
- All the possible locations of the containers in the cargo holds (see for example Figure 1).
- A given subset, I, of the container list that the user wishes to be loaded (containers that cannot be left on the ground because they contain e.g. perishable goods)—will determine the freight constraints (7) in what follows.
- A list of couples (i,k) for any given container i that the user wants to be in a specific compartment k (for instance near a door to be landed at a stopover, or for toxic material containers to be away from the foodstuff container hold, etc.) Moreover, this allows for the possibility of requiring some given containers, already loaded by the ground crew, not to be displaced, in order to expedite the plane's departure—will determine the freight constraints (8).

B. Optimization variables (unknowns)

The decision variables are binary: $x_{ij} \in \{0, 1\}$ is 1 if container i is to be placed in compartment j, and 0 otherwise $(i = 1, 2, \ldots, N_{cont};\ j = 1, 2, \ldots, N_{comp})$.

C. Output

The purpose of the method is producing a list of containers to be loaded in each compartment, plus a list of containers that are to be left on the ground.

D. Further notation

Two critical quantities, which both depend upon the vector x of decision variables x_{ij} defined above, are:

- *Total mass loaded*:

$$M(x) := \sum_{i=1}^{N_{cont}} \sum_{j=1}^{N_{comp}} M_i x_{ij}; \tag{1}$$

- *Center of gravity of the aircraft after loading*:

$$CG(x) := \frac{M_a X_a + \sum_{i=1}^{N_{cont}} \sum_{j=1}^{N_{comp}} M_i X_j x_{ij}}{M_a + M(x)}. \tag{2}$$

E. Constraints

- **Aircraft constraints**:

– Stability requirements:

$$CG(x) \leq X_{\text{stab}}. \tag{3}$$

– Stress/mass capacity constraints (overall and for each hold):

$$M(x) \leq M_{\text{max}}; \tag{4}$$

$$\sum_{i=1}^{N_{cont}} \sum_{j \in H_k} M_i x_{ij} \leq M^k_{\text{max}}, \quad k = 1, 2, \ldots, N_{hold}. \tag{5}$$

– Volume capacity constraints: A priori combinatorial and nonlinear in nature, we defer their mathematical formulation to Section III. We shall demonstrate there, on a specific instance, how to model volume capacity constraints in the framework of an integer *linear* programming formulation, through the addition of a small number of decision variables.

- **Mathematical constraints**:

– Each container must be loaded at most once. By the binary definition of the x_{ij}'s, this can equivalently simply be written as:

$$\sum_{j=1}^{N_{comp}} x_{ij} \leq 1, \quad \text{for all } i = 1, 2, \ldots, N_{cont}. \tag{6}$$

- **Freight constraints**:

As specified in the input list of Section II-A:

– If a subset, I, of the container list is required to be loaded:

$$\sum_{j=1}^{N_{comp}} x_{ij} = 1, \quad \text{for all } i \in I. \tag{7}$$

– For any given container i required to be in a specific compartment k:

$$x_{ik} = 1 \tag{8}$$

(and therefore, by (6), $x_{ij} = 0$, for all $j \neq k$).

F. Objective of the optimization problem

Remember that we are faced with two contradictory objectives: maximizing $M(x)$, and having $CG(x)$ as far aft as possible (but not behind the limit imposed by stability requirements—Constraint (3)). One way to proceed, with an optimization approach, would be to consider the (longitudinal) position of the center of gravity of the aircraft as the objective function to be maximized, subject to the constraint of loading at least some pre-specified mass of freight. The problem with this approach is twofold. Firstly, the total freight mass loaded is not optimized. Secondly, the objective function, $CG(x)$, is a *nonlinear* function of the optimization variables x_{ij}'s. The difficulty of solving an integer nonlinear programming problem is incomparably higher than that of solving an integer *linear* programming problem. One cannot reasonably expect to be able to obtain the optimal solution of an integer nonlinear programming problem. Indeed, viable methods for solving general integer nonlinear programming problems include stochastic methods (such as simulated annealing, genetic algorithms, and tabu search), which provide no guarantee of getting close to optimality, and deterministic methods (such as arborescent methods—branch and bound, for instance). To give an idea of the extra difficulty involved in presence of nonlinearities, let us consider the most widely used deterministic method for solving integer programming problems: branch and bound. In order to be efficient, branch-and-bound procedures must find, as subproblems, globally optimal solutions of continuous relaxations of the problem. In the case of an integer *linear* programming formulation, this subproblem is a simple linear programming problem, whereas it is an extremely difficult problem in the case of an integer *nonlinear* programming formulation. However, in the case where one assumes that the complete given list of containers must be loaded, the numerator of (2) is then a constant, and $CG(x)$ is therefore linear. This is a rather strong assumption for practical problems. This indeed means that one chooses a priori which containers are to be loaded, and assumes that the containers chosen do fit in the aircraft. This way of operating is clearly likely to yield solutions which are not optimal with respect to both the centering *and* the mass criteria. Such practical considerations motivate our method, which does not rely on the knowledge of which of the containers are to be loaded. We propose **maximizing the mass loaded**, given by (1), subject to the the constraints (3) to (8), plus the following additional constraint.

G. Centering constraint

- Keeping the center of gravity within a reasonable distance from the ideal position:

$$X_{\text{target}} - \epsilon \leq CG(x) \leq X_{\text{target}} + \epsilon, \tag{9}$$

where ϵ is some positive allowable displacement of the center of gravity of the aircraft from its ideal position. The value of ϵ is set by the airliner in such a way as to account for uncertainties in the geometric and weight data. This constraint can equivalently be written as:

$$[M_a + M(x)][X_{\text{target}} - \epsilon] \leq M_a X_a + \sum_{i=1}^{N_{cont}} \sum_{j=1}^{N_{comp}} M_i X_j x_{ij} \leq [M_a + M(x)][X_{\text{target}} + \epsilon], \tag{10}$$

i.e. it can be expressed as two linear (inequality) constraints.

All of the above constraints, (3) to (8), (10), and the objective function (1), are *linear*. Moreover, in Section III, we shall also model linearly the volume capacity constraints. Thus, the resulting mathematical formulation will be an integer *linear* programming problem. We shall see that the number of integer variables involved will be small enough so that the problem will be easy to solve with off-the-shelf software. The purpose of the next section is therefore to demonstrate how the volume capacity constraints can be modeled so as to fit within the integer linear programming formulation.

III. Modeling volume capacity constraints

The way we model the volume capacity constraints is specific to each type of aircraft and to the different types of containers one has to load (note that the given list of containers will typically include several different types of containers). In order to simplify the presentation, and for illustrative purposes, we shall consider here a special instance: the Airbus A340-300 together with five different types of containers, described in Table I. Indeed, as we shall see, modeling volume capacity constraints involve non-convex piecewise-linear functions. Such constraints cannot be dealt with directly by off-the-shelf optimization software. We propose a way to transform such constraints into simple linear constraints, with the introduction of a small number of extra integer variables (for choosing between subdomains over which the function is convex). The methodology we are about to present can nevertheless straightforwardly be applied to any other commercial carrier in an analogous manner. Note that in practice, these volume capacity constraints are to be generated a priori once and for all, for each type of aircraft owned by the airliner.

Figure 1 displays the different homogeneous arrangements of each of the possible types of container in the forward (compartments 1 and 2) and aft (compartments 3 and 4) cargo holds of the Airbus A340-300 aircraft. We partition the five types of containers into three categories, according to the

TABLE I

THE DIFFERENT TYPES OF CONTAINERS CONSIDERED

<table>
<tr><th>Type</th><th>Description</th><th>Space required</th></tr>
<tr><td>1</td><td>Half size
NAS 3610-2K1C, 2K2C
60.4 x 61.5 inches
Max IATA contour E, G</td><td>1 small place</td></tr>
<tr><td>2</td><td>Half size
NAS 3610-2K1C, 2K2C
60.4 x 61.5 inches
Max IATA contour C, H</td><td rowspan="2">2 small places</td></tr>
<tr><td>3</td><td>Full size
NAS 3610-2L1C, 2L2C
60.4 x 125 inches
Max IATA contour F</td></tr>
<tr><td>4</td><td>Full size
NAS 3610-2A2C, 2A6C
88 x 125 inches
Max IATA contour F</td><td rowspan="2">1 large place</td></tr>
<tr><td>5</td><td>Full size
NAS 3610-2M1C, 2M3C
96 x 125 inches
Max IATA contour F</td></tr>
</table>

space one container occupies in a hold: We note that containers of types 2 and 3 require twice as much space in the hold as one container of type 1. We state that the space occupied by a container of type 1 is a *small place*. A container of type 2 or 3 therefore requires two small places each. Analogously, we note that containers of types 4 and 5 both occupy the same space, which we call a *large place* (it is not an integer multiple of a small place). This is summarized in Table I.

There are many ways of combining small and large places within a given compartment. It will be convenient to define, for each compartment j ($j = 1, 2, \ldots, N_{comp}$):

s_j number of small places occupied in compartment j

l_j number of large places occupied in compartment j.

Since for each container i ($i = 1, 2, \ldots, N_{cont}$), we are given its type as input:

T_i ($\in \{1, 2, 3, 4, 5\}$) type of container i,

we have, in terms of the optimization variables x_{ij}'s:

$$s_j := \sum_{i:T_i=1} x_{ij} + 2(\sum_{i:T_i=2} x_{ij} + \sum_{i:T_i=3} x_{ij}), \tag{11}$$

$$l_j := \sum_{i:T_i=4} x_{ij} + \sum_{i:T_i=5} x_{ij} \tag{12}$$

(remember that a containers of category 2 or 3 requires *two* small places in the holds). Note that the s_j's and l_j's are linear combinations of the optimization variables.

In what follows, for each compartment, we shall initially model the volume capacity constraints with logical constraints (alternative/conditional sets of constraints), and we shall, afterwards, express these constraints so as they fit in our integer linear programming formulation, following modeling techniques described, for example, in [14, Section 9.2].

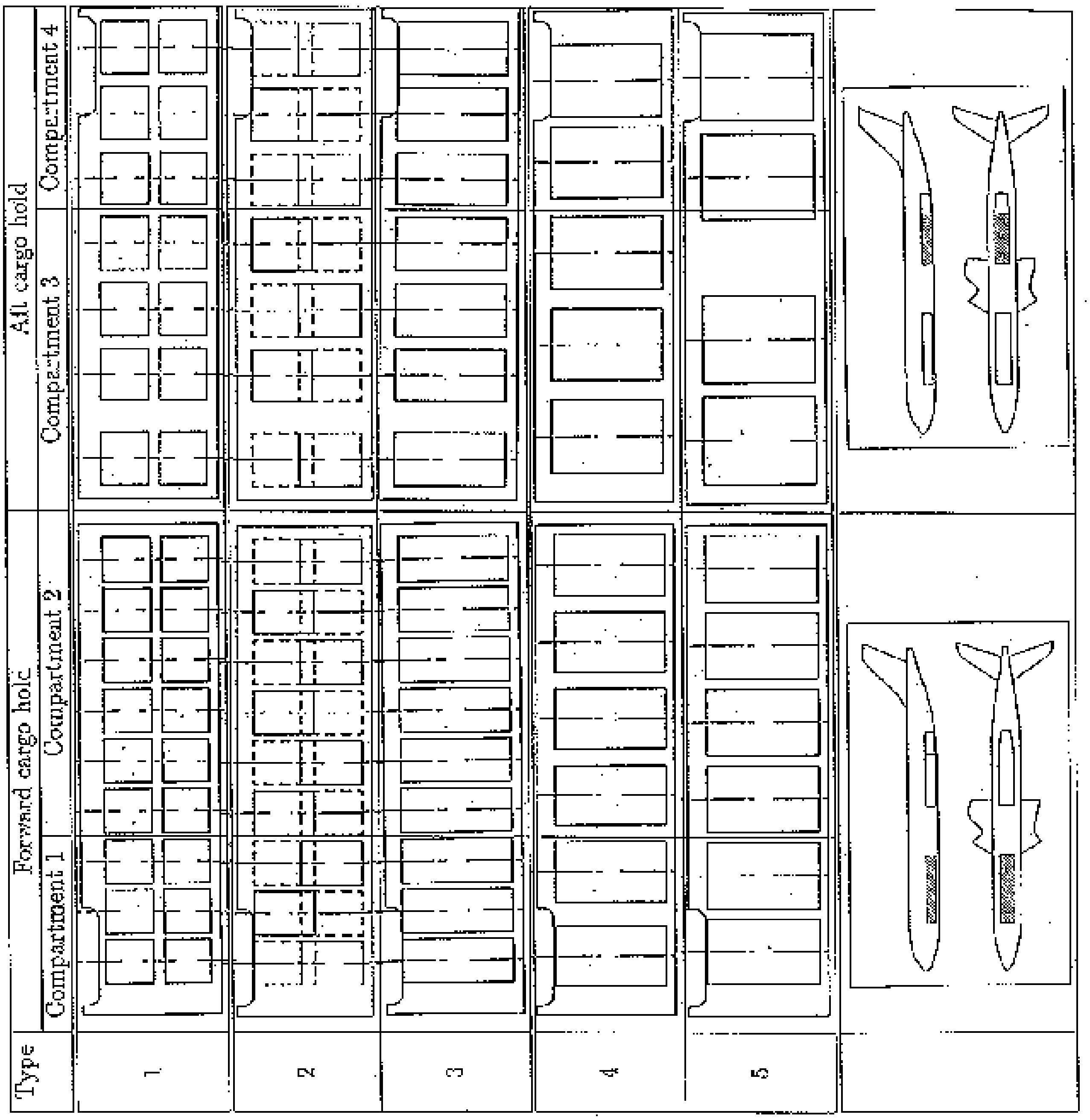

Fig. 1. Arrangements of containers in the Airbus A340-300 holds

A. Compartment 1

Inspection of Figure 1 reveals that one can load in compartment 1 any combinations of containers requiring up to a total of 6 small places and 0 large place. One can alternatively fit containers requiring up to a total of 2 small places and 1 large place in compartment 1. A third alternative is: 0 small place and 2 large places. The squares displayed on Figure 2 summarize all the possible combinations. These possible arrangements cannot be modeled through a *single* set of inequalities involving linearly the integer optimization variables x_{ij}'s. We could consider here, separately, the three alternatives l_1 = 0, 1, or 2. However, we shall try as much as possible to model the relationship between the number of small places and large places in each of the compartments, with the fewest possible alternatives. The best we can do here is to consider separately *two* possibilities. There is indeed no way to find *one*

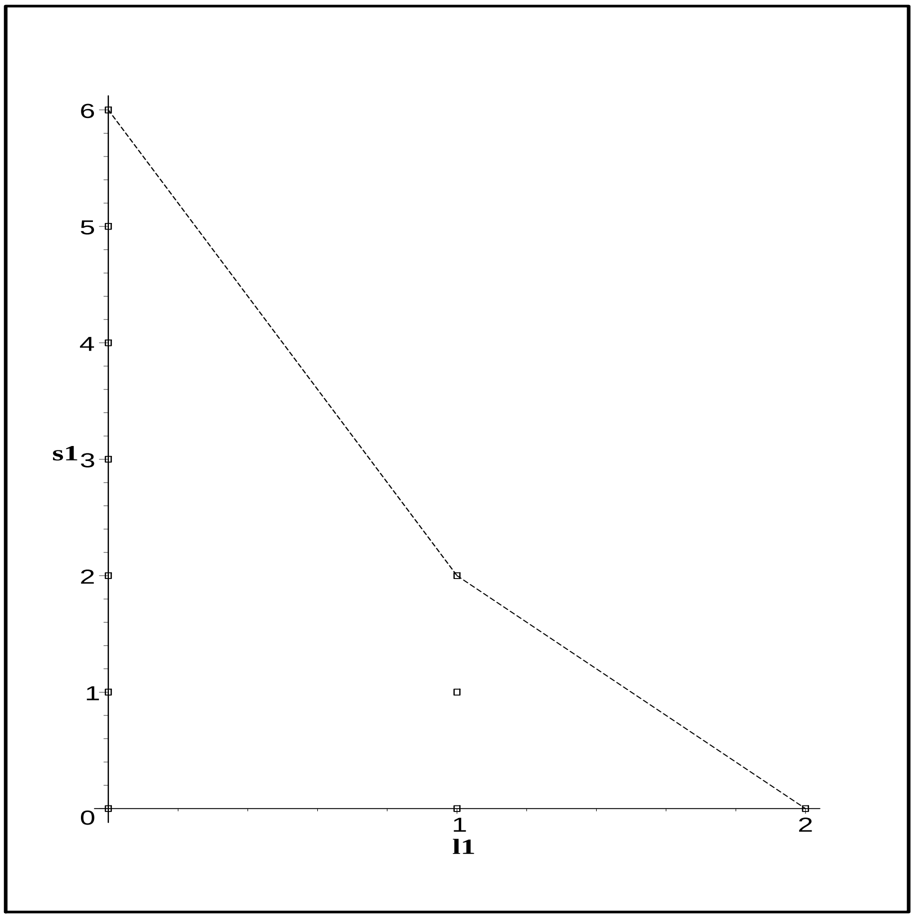

Fig. 2. Possible arrangements in compartment 1

system of linear inequalities (which would thus define a convex set) which would include all the feasible points (represented by squares) but not the undesirable (because infeasible) point: $(l_1 = 1, s_1 = 3)$. We choose to consider separately the following two alternatives: $l_1 \leq 1$ and $l_2 = 2$ (remember that by the binary definition of the x_{ij}'s, the l_j's and the s_j's are bounded to be non-negative and integer valued—cf. equations (11) and (12)). The above enumeration of combinations of small and large places for loading compartment 1 can equivalently then simply be represented by :

$$s_1 + 4l_1 \leq 6, \quad \text{if } l_1 \leq 1, \tag{13}$$

$$s_1 = 0, \quad \text{if } l_1 = 2, \tag{14}$$

(together with the binary definition of the decision variables: $x_{ij} \in \{0, 1\}$). We next show how the two alternatives (13) and (14) can be expressed so as to fit within an integer linear programming formulation.

Constraints (13) and (14) can equivalently be rewritten as the two alternatives (*disjunctive constraints*):

$$(s_1 + 4l_1 \leq 6, \quad \text{and} \quad l_1 \leq 1) \quad \text{or} \quad (s_1 \leq 0, \quad \text{and} \quad l_1 = 2).$$

Let us now introduce an extra binary variable, $y_{1,1}$, having value zero when the first alternative holds and value one corresponding to the second alternative. Hence, (13) and (14) can be written as:

$$\left\{\begin{array}{rl} s_1 + 4l_1 - B_{1,1}y_{1,1} & \leq 6, \\ l_1 - B_{1,2}y_{1,1} & \leq 1, \\ s_1 - B_{1,3}(1 - y_{1,1}) & \leq 0, \\ -l_1 - B_{1,4}(1 - y_{1,1}) & \leq -2, \\ l_1 - B_{1,5}(1 - y_{1,1}) & \leq 2, \end{array}\right. \tag{15}$$

where we introduced five “big enough” constants $B_{1,i}$. Each constant $B_{1,i}$ is to be set in such a way that the corresponding inequality is always satisfied for any feasible solution to our problem as soon as the coefficient of $B_{1,i}$ is one. This can easily be done. Consider, for instance, the first inequality. Since $s_1 + 4l_1$ can clearly never be above 14, it suffices to set $B_{1,1} := 8$. Hence, when $y_{1,1} = 1$, the first constraint is not restrictive (since always satisfied), only the last three constraints are relevant. Similarly, it suffices for instance to set $B_{1,2} := 1$, $B_{1,3} := 6$, $B_{1,4} := 2$, and $B_{1,5} := 4$.

Naturally, all the variables s_i's and l_i's can be replaced by appropriate x_{ij}'s using (11) and (12). We thus obtain a set of linear constraints—linear in the binary variables x_{ij}, $y_{1,1}$—after adding the extra binary variable $y_{1,1}$.

B. Compartment 2

Again, from inspection of Figure 1, we obtain, in an analogous manner, Figure 3, which displays all the possible combinations of small places and large places in compartment 2. Hence, volume capacity

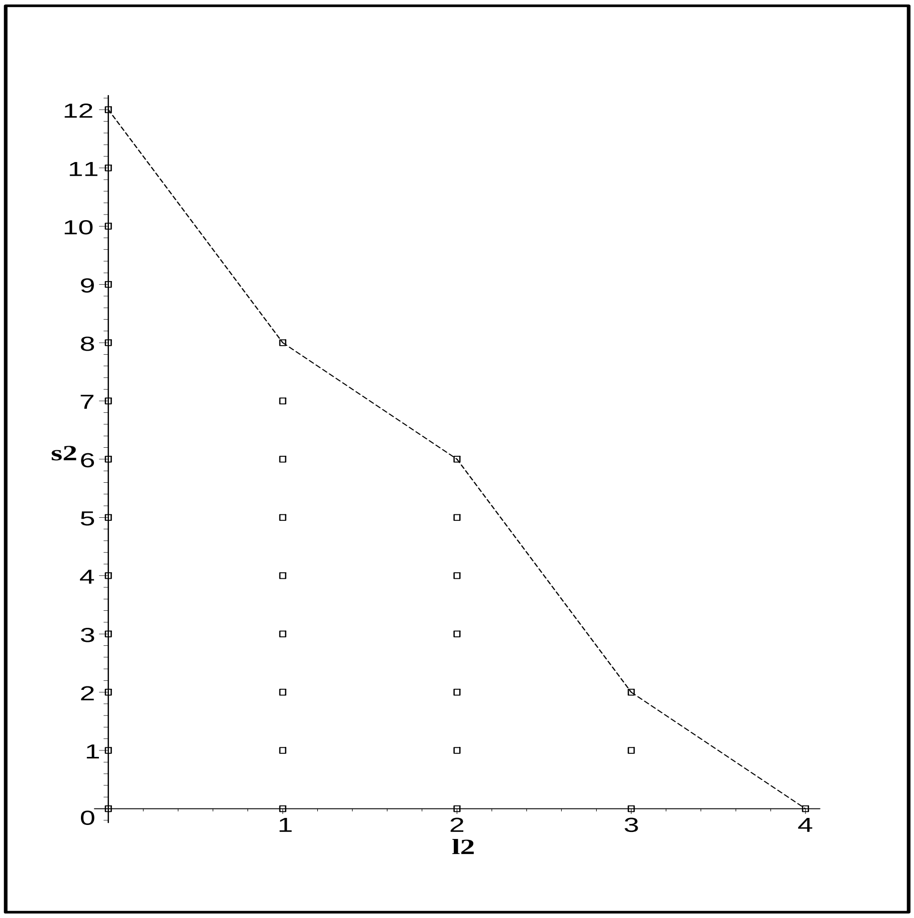

Fig. 3. Possible arrangements in compartment 2

constraints corresponding to compartment 2 are as follows. We have to consider a minimum of three alternatives. We choose here:

$$s_2 + 4l_2 \leq 12, \quad \text{if } l_2 \leq 1, \tag{16}$$
$$s_2 + 4l_2 \leq 14, \quad \text{if } 2 \leq l_2 \leq 3, \tag{17}$$
$$s_2 = 0, \quad \text{if } l_2 = 4. \tag{18}$$

As we did for compartment 1, we next introduce extra binary variables $y_{2,i} \in \{0,1\}$, $i = 1, 2, 3$, which monitor which of the three alternatives is relevant. Having $y_{2,i} = 0$ means that the (in)equalities of the ith alternative must be satisfied. We can then replace (16), (17), and (18) with:

$$\left\{ \begin{array}{rl} s_2 + 4l_2 - B_{2,1}y_{2,1} & \leq 12, \\ l_2 - B_{2,2}y_{2,1} & \leq 1, \\ s_2 + 4l_2 - B_{2,3}y_{2,2} & \leq 14, \\ -l_2 - B_{2,4}y_{2,2} & \leq -2, \\ l_2 - B_{2,5}y_{2,2} & \leq 3, \\ s_2 - B_{2,6}y_{2,3} & \leq 0, \\ -l_2 - B_{2,7}y_{2,3} & \leq -4, \\ l_2 - B_{2,8}y_{2,3} & \leq 4, \\ y_{2,1} + y_{2,2} + y_{2,3} & = 2, \end{array} \right. \tag{19}$$

where the $B_{1,i}$'s are appropriately initialized "big enough" constants (set e.g. $B_{2,1} := 16$, $B_{2,2} := 3$, $B_{2,3} := 14$, $B_{2,4} := 2$, $B_{2,5} := 1$, $B_{2,6} := 4$, $B_{2,7} := 4$, and $B_{2,8} := 0$). The last equality ensures that one of the $y_{1,i}$'s is zero. Note that, in fact, only two extra binary variables $y_{1,i}$'s suffice, as we can eliminate the last equality constraint in order to replace $y_{2,3}$ everywhere with $2 - y_{2,1} - y_{2,2}$. We thus modeled the volume capacity constraints for compartment 2 with a set of linear constraints after adding two more binary variables: $y_{2,1}$ and $y_{2,2}$.

C. Compartment 3

Firstly, it is clear from Figure 1 that our model must include the constraint

- forbidding more than 2 containers of type 5 in compartment 3:

$$\sum_{i:T_i=5} x_{i3} \leq 2. \tag{20}$$

We next take into account the possible overlapping of a container of compartment 4 over compartment 3. Inspection of Figure 1 indeed reveals that a second container of type 5 in compartment 4 overlaps slightly compartment 3 (cf. the dashed line on Figure 1): it therefore does not leave more than 2 large places in compartment 3, and not more than 6 small places in compartment 3 if such an overlap occurs. We first consider separately whether there is an overlapping container or not, i.e. whether the number of containers of type 5 in compartment 4 is equal to two or not:

1. If $\sum_{i:T_i=5} x_{i4} = 2$ (overlap), then all the ways of combining small and large places within compartment 3 are represented (again by squares) in Figure 4. It is possible here to model these possible arrangements with a *single* set of inequalities:

$$s_3 + 2l_3 \leq 6, \tag{21}$$
$$s_3 + 4l_3 \leq 8. \tag{22}$$

2. If $\sum_{i:T_i=5} x_{i4} \leq 1$ (no overlap), we then obtain Figure 5, and subsequently we must consider further

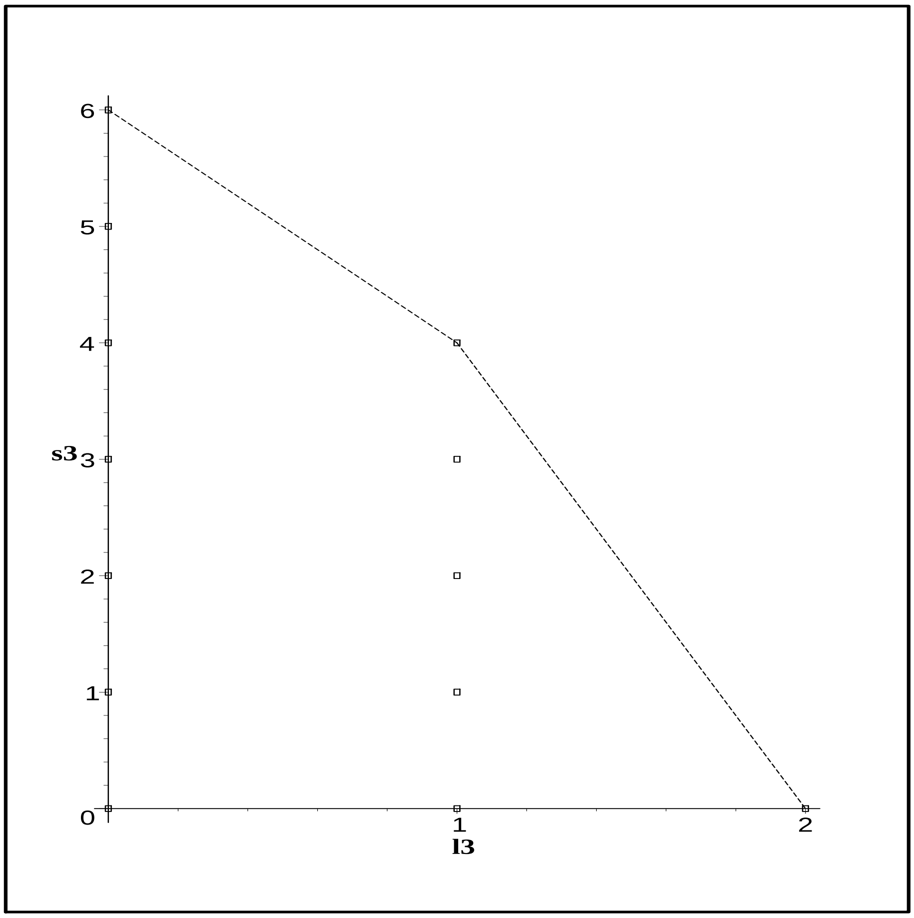

Fig. 4. Possible arrangements in compartment 3 with overlap

two alternative sets of constraints:

$$s_3 + 2l_3 \leq 8, \quad \text{if } l_3 \leq 1, \tag{23}$$

$$s_3 + 2l_3 \leq 6, \quad \text{if } l_3 \geq 2. \tag{24}$$

We can replace (21), (22), (23), and (24) in an analogous manner as before with:

$$\left\{ \begin{array}{rl} -\sum\limits_{i:T_i=5} x_{i4} - B_{3,1}y_{3,1} & \leq -2, \\ s_3 + 2l_3 - B_{3,2}y_{3,1} & \leq 6, \\ s_3 + 4l_3 - B_{3,3}y_{3,1} & \leq 8, \\ \sum\limits_{i:T_i=5} x_{i4} - B_{3,4}y_{3,2} & \leq 1, \\ s_3 + 2l_3 - B_{3,5}y_{3,2} & \leq 8, \\ l_3 - B_{3,6}y_{3,2} & \leq 1, \\ \sum\limits_{i:T_i=5} x_{i4} - B_{3,7}y_{3,3} & \leq 1, \\ s_3 + 2l_3 - B_{3,8}y_{3,3} & \leq 6, \\ -l_3 - B_{3,9}y_{3,3} & \leq -2, \\ l_3 - B_{3,9}y_{3,3} & \leq 2, \\ y_{3,1} + y_{3,2} + y_{3,3} & = 2, \end{array} \right. \tag{25}$$

after introducing three (two suffice) extra binary variables $y_{3,i}$, and appropriate constants $B_{3,i}$ (set e.g. $B_{3,1} := 2$, $B_{3,2} := 8$, $B_{3,3} := 12$, $B_{3,4} := 1$, $B_{3,5} := 6$, $B_{3,6} := 2$, $B_{3,7} := 1$, $B_{3,8} := 8$, $B_{3,9} := 2$,

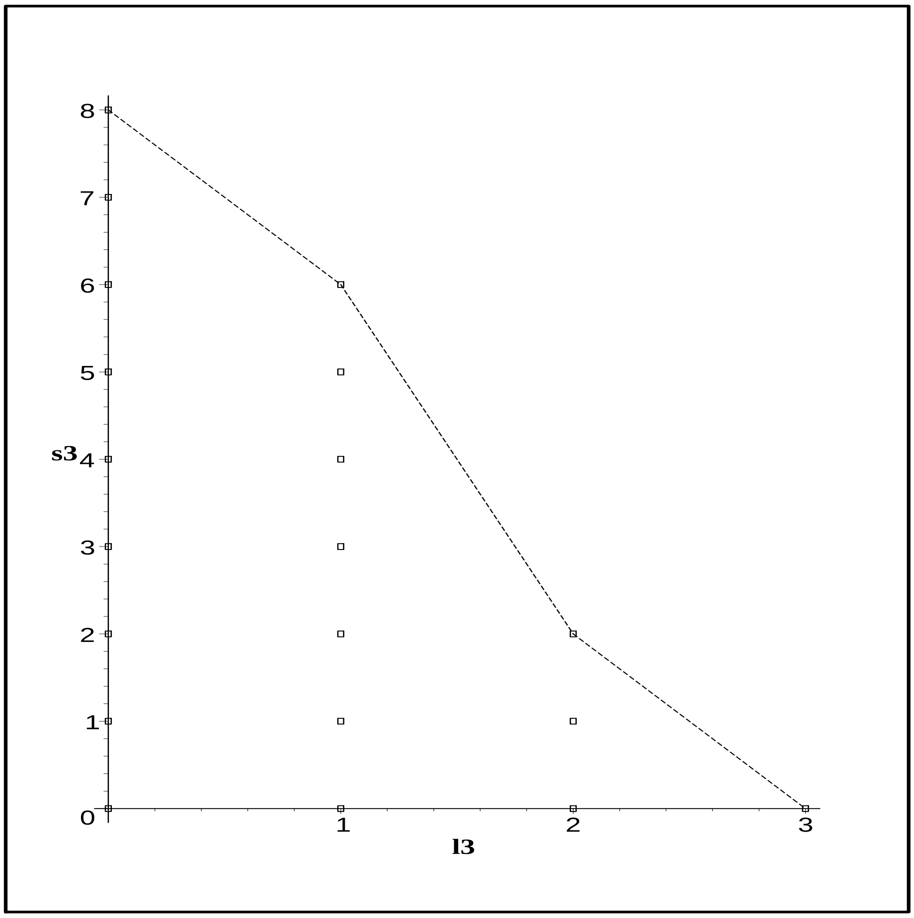

Fig. 5. Possible arrangements in compartment 3 without overlap

and $B_{3,9} := 1$). The first three above constraints corresponds to the system (21)–(22), the next three corresponds to (23), and the last three to (24).

D. Compartment 4

The analysis for compartment 4 is identical to that for compartment 1:

$$\begin{aligned} s_4 + 4l_4 &\leq 6, \quad \text{if } l_4 \leq 1, \\ s_4 &= 0, \quad \text{if } l_4 = 2, \end{aligned}$$

which yields the set of linear constraints

$$\left\{ \begin{array}{rl} s_4 + 4l_4 - B_{4,1}y_{4,1} & \leq 6, \\ l_4 - B_{4,2}y_{4,1} & \leq 1, \\ s_4 - B_{4,3}(1 - y_{4,1)} & \leq 0, \\ -l_4 - B_{4,4}(1 - y_{4,1}) & \leq -2, \\ l_4 - B_{4,5}(1 - y_{4,1}) & \leq 2, \end{array} \right. \tag{26}$$

after adding the extra binary variable $y_{4,1}$ (it suffices to set $B_{4,1} := 8$, $B_{4,2} := 1$, $B_{4,3} := 6$, $B_{4,4} := 2$, and $B_{4,5} := 4$).

IV. Overall formulation

The volume capacity constraints of the previous section are relevant to the specific instance of the Airbus A340-300 aircraft together with the five different types of containers described in Table I.

However, the procedure we demonstrated on a particular case can straightforwardly be applied to another commercial carrier and/or with other types of containers, following the same lines as in Section III. For the example we study in the current paper, the volume capacity constraints required the addition of six extra binary variables (the y's).

To summarize, here is the overall integer linear programming formulation of the aircraft container loading problem:

$$\begin{array}{ll} \text{maximize} & M(x), \text{ given by (1)} \\ \text{subject to} & (3),(4),(5),(6),(7),(8),(10),(15),(19),(20),(25),(26) \end{array}$$

where

- the optimization variables are all binary: the x_{ij}'s, plus the six y_{kl}'s (which appear only in the constraints—not in the objective function) introduced in the previous section for deciding amongst various alternatives;
- the objective function and the constraints are linear in the components of x and y.

V. Numerical Results

In the previous sections, we achieved modeling the aircraft loading problem with an integer linear programming formulation. This allows the direct use of off-the-shelf software for solving the problem. We first build up 6 test problems (6 lists of containers of different types and various masses to be loaded in an Airbus A340-300 aircraft), as described in Table II. The first test problem involves simply 40

TABLE II

The 6 test problems

Test Problem	Total Mass (Kg)	Total number of containers	Number of containers of type:			
			1	2 & 3	4	5
A	40 000	40	10	10	10	10
B	49 060	40	10	10	10	10
C	48 408	40	10	10	10	10
D	36 610	22	14	0	6	2
E	39 926	26	14	6	6	0
F	29 890	21	14	1	6	0

containers of various types but with identical weight: 1 000 Kg. Test problem B has 40 containers of different masses which differ only slightly from one another. The 40 containers of Test problem C are of various masses. These first three test problems include a number of containers for which it is easy to see *a priori* that the whole list cannot be loaded. Test Problems D, E and F, which involve respectively 22, 26 and 21 containers, are real-life problems. We chose to use directly NAG's branch-and-bound based subroutine H02BBF for integer linear programming in order to solve the problem (one can equivalently consider using CPLEX for this purpose). The computing times reported here are from experimenting on a Pentium 100 MHz (RAM: 64 Mo, disk of 1.2 Go) under Windows NT 4.0. The first three test problems (Problems A, B, and C) served to tune the user-defined parameters of the NAG subroutine. These were accordingly set, for all the computational results we are presenting, to: itmax = 500, maxnod = 500, toliv = 10^{-5}, tolfes = machine epsilon. We express the (longitudinal) position of the center of gravity in terms of percentage of the *reference chord* (RC). For the Airbus A340-300 aircraft, RC = 7.27 meters. For our tests the ideal centering is 30% RC aft. We performed two series of tests. One, with a tolerated deviation from the ideal centering of $\epsilon = 6\%$ RC (which corresponds to 43.62, cm for the aircraft under study) in order to stay within operational limits, as is common practice for airliners. The second series is with a maximal allowed error of $\epsilon = 1\%$ RC (if one

TABLE III
NUMERICAL RESULTS

Max allowed error ϵ (% RC)	Test prob	Nb of containers loaded	Mass loaded (Kg)	Centering error obtained (% RC)	CPU (min)
6	A	21/40	21 000	5.49	2.9
	B	21/40	25 510	5.88	3.9
	C	16/40	22 910	5.98	6.3
	D	19/22	31 990	5.67	4.0
	E	21/26	30 090	3.64	3.2
	F	21/21	29 890	5.93	4.0
1	A	20/40	20 000	0.14	5.1
	B	20/40	24 307	0.24	9.9
	C	16/40	19 890	0.97	6.6
	D	20/22	31 690	1.00	3.5
	E	22/26	29 836	0.97	6.0
	F	20/21	28 890	0.49	6.0

favours further fuel saving). Table III summarizes our computational results. Note that these CPU times correspond to *optimal* solutions. (Remember that, except for test problem A, the containers are of various masses, it is therefore not surprising that for test problem D, we load fewer containers with an allowed error of $\epsilon = 6\%$ RC than with $\epsilon = 1\%$ RC: in the former case the mass of the containers loaded is *greater* than in the latter case.)

The CPU times reported show that our approach is viable for practical use. Note that a partial branch-and-bound resolution could yield feasible loadings even more rapidly. Indeed, ground personnel naturally does not have to wait for the process to be over in order to implement some intermediate satisfying sub-optimal solution generated by the program. Note that in practice, our method can recursively be invoked whenever the centering required (through the parameter ϵ) happens to be too tight (either the obtained loaded mass appears to be insufficient, or the current ϵ yields an infeasible problem). Finally, note that in the above-reported numerical results, no freight constraints of type (7) and (8) were present. However, we did try our method with such constraints, and the CPU times obtained generally only improved (not surprisingly, as adding constraints further restricts the discrete search space).

VI. Conclusions

We considered, in this paper, the aircraft container loading problem, more specifically the problem of choosing which containers should be loaded on the aircraft, and how they should be distributed amongst the different compartments, in order to improve fuel consumption while optimizing freight income, subject to structural and safety constraints. We restricted our study to the case of long-range aircrafts (Airbus A340-300), which allowed approximating the domain of possible positions of the center of gravity, in terms of the aircraft gross weight, by a box (upper- and lower-bounds constraints). We avoided thereby nonlinear constraints in our formulation. Future work could address the more general problem in which this domain, of possible positions of the center of gravity, is a polyhedron.

We described a way to model volume capacity constraints, which involved non-convex piecewise-linear functions, with linear constraints by adding a small number of binary variables. Moreover, the integer linear programming formulation we introduced straightforwardly enables the integration of various constraints of practical relevance. For the preliminary tests we presented, we used an off-the-shelf integer linear programming solver on a PC. The acceptable computer times required to obtain *optimal* solutions on some real-life test problems show that the approach is viable in practical situations.

Possible extensions of our work, which could be interesting from the airliner point of view, include the following. Using different values for the centering tolerance (parameter ϵ), one could attempt to balance the importance of the two conflicting objectives (freight loaded versus fuel consumption). The approach we introduced in this paper could indeed be used in order to generate different feasible solutions to the problem (a set of proposed positions of the center of gravity, and corresponding optimal mass loaded). Further, one could consider implementing a procedure which automatically chooses which solution is the best amongst the ones proposed in terms of operational cost, knowing the income expected per loaded ton of freight, the impact in terms of fuel saving for each centimeter of displacement of the center of gravity on a specific aircraft, and the current cost of fuel. Finally, note that airliners often do not pay much attention to the aircraft container loading problem because of the possibility of using fuel transfer systems to improve the position of the center of gravity. Further work could also consider exploiting such systems within our optimization approach in order to allow more flexibility, improving thereby results (the potential of the sole use of fuel transfer systems is indeed limited as the aircraft approaches its destination and less fuel is available to transfer).

References

[1] S. V. Amiouny, J. J. Bartholdi III, J. H. Vande Vate, and J. Zhang, "Balanced loading," *Operations Research*, vol. 40, no. 2, pp. 239–246, 1992.

[2] L. A. Martin-Vega, "Aircraft load planning and the computer: Description and review," *Computers & Industrial Engineering*, vol. 9, no. 4, pp. 357–369, 1985.

[3] O. Larsen and G. Mikkelsen, "An interactive system for the loading of cargo aircraft," *European Journal of Operational Research*, vol. 4, pp. 367–373, 1980.

[4] M. S. Frankel *et al.*, "Army data distributions system/packet radio testbed, Contract MDA903-80-C-0217," Quaterly Technical Reports 1980–81, SRI International, Menlo Park, California 94025.

[5] K. R. Heidelberg, G. S. Parnell, and J. F. Ames IV, "Automated air load planning," *Naval Research Logistics*, vol. 45, no. 8, pp. 751–768, 1998.

[6] D. D. Cochard and K. A. Yost, "Improving utilization of air force cargo aircraft," *Interfaces*, vol. 15, pp. 53–68, 1985.

[7] I. Brosh, "Optimal cargo allocation on board a plane: a sequential linear programming approach," *European Journal of Operational Research*, vol. 8, pp. 40–46, 1981.

[8] K. Y. K. Ng, "A multicriteria optimization approach to aircraft loading," *Operations Research*, vol. 40, no. 6, pp. 1200–1205, 1992.

[9] K. Mathur, "An integer-programming-based heuristic for the balanced loading problem," *Operations Research Letters*, vol. 22, pp. 19–25, 1998.

[10] C. Thomas, K. Campbell, G. Hines, and M. Racer, "Airbus packing at Federal Express," *Interfaces*, vol. 28, no. 4, pp. 21–30, 1998.

[11] A. P. Davies and E. E. Bischoff, "Weight distribution considerations in cargo loading," *European Journal of Operational Research*, vol. 114, no. 3, pp. 509–527, 1999.

[12] N. Boseel and P. Dejax, "Models and algorithms for container allocation problems on trains in a rapid transshipment shunting yard," *Transportation Science*, vol. 32, no. 4, pp. 370–379, 1998.

[13] Lai Lai Tung and E. Turban, "Expert systems support container operations in the Port of Singapore," *New Review of Applied Expert Systems*, vol. 2, pp. 63–75, 1996.

[14] S. P. Bradley, A. C. Hax, and T. L. Magnanti, *Applied Mathematical Programming*, Addison-Wesley, 1977.

Marcel Mongeau received his BSc (1985) and MSc (1987) degrees in Mathematics from the Université de Montréal, and his PhD (1991) in Combinatorics & Optimization from the University of Waterloo. He was then post-doctoral researcher at the Centre de Recherches Mathématiques (Université de Montréal), and at the Institut National de la Recherche en Informatique et en Automatique (Rocquencourt, France). In 1993-94, he was Research Fellow at the University of Edinburgh, where he worked on applications of optimization in chemical engineering. He is currently Maître de Conférences in Mathematics at the Université Paul Sabatier (Toulouse, France) since 1994. His research interests include global optimization, numerical optimization, and operations research.

Christian Bès received a doctorate in automatic from the Université Paul Sabatier (Toulouse, France) in 1984. From 1984 to 1986, he worked on operations research as a post-doctoral researcher at the University of Toronto, and at the Institut National de la Recherche en Informatique et en Automatique (Rocquencourt, France). He received an *Habilitation à Diriger des Recherches* in Applied Mathematics from the Université Paul Sabatier in 1995. He worked with Airbus France in Toulouse from 1986 to 1999. He is currently Professeur in Mechanical Engineering at the Université Paul Sabatier. His current research interests include mechanical design and multidisciplinary optimization.