The Container Loading Problem with Tipping Considerations

Robert H. Storer
Joseph C. Hartman
Lehigh University

Report No. 00T-010

The Container Loading Problem with Tipping Considerations

Robert H. Storer and Joseph C. Hartman
Industrial and Manufacturing Systems Engineering
Lehigh University

Abstract

The container loading problem is defined as follows: assign a set of N containers, differing in weight but equal in size, to N positions on an airplane that locates the center of mass along the "front to back" axis to a pre-specified point. We provide both an exact method and an effective heuristic to solve the problem. The solutions provided by the heuristic are quite good, and certainly within the accuracy of the measurement of weights, as shown by simulation. Additionally, as containers are loaded from the back to front, we address the possibility of tipping the plane during loading. We analyze conditions that must be met to prevent tipping and provide a modified approach to our algorithm if these conditions are not met.

1. Introduction

We investigate problems surrounding the loading of cargo containers onto airplanes that arise in overnight package delivery firms. The basic procedure involves first loading containers with packages, weighing the loaded containers, then pulling the containers out to the flight deck to be loaded on the plane. All containers are the same size, and are shaped so as to fill the entire cross-section of the plane. Containers are loaded through a door at the front of the plane and pushed to the back where they are secured for take-off. Thus the i^{th} container loaded will be located in the i^{th} position from the rear of the plane.

There are two important considerations when determining the container loading sequence (and thus the positions the containers occupy on the plane). First, if containers loaded early in the sequence are too heavy, the plane may actually tip over backwards, an occurrence that clearly must be avoided (but which has happened in practice). The second involves the location of the center of mass (*cm*) of the fully loaded plane with respect to the "front to back" axis. Ideally, the *cm* is located such that the plane is balanced when flying. If the *cm* is off a bit in either direction, the plane can still fly, but must use "flaps" to compensate for the imbalance. This compensation requires greater fuel consumption, thus there is an increasing cost penalty associated with increasing imbalance.

We study the static version of this problem in this paper. That is, we assume that all containers are weighed and available before loading begins. We further assume that when each container is loaded, it is immediately pushed to the back of the plane and secured. Thus, the critical containers with respect to tipping are those loaded first.

Previous research has looked at various problems in airplane loading. Martin-Vega (1985) investigates dividing cargo into groups with one group being assigned to each plane. However, he does not address how each plane is to be loaded. The issue of packing a plane's cargo deck is basically a two-dimensional bin packing problem for which a significant literature exists. However the issue of balance is not addressed in the two-dimensional bin packing literature. Cochard and Yost (1985) develop heuristics within a decision support framework to aid load planners in cargo loading. They first address the two-dimensional bin packing problem with heuristics, then seek to balance the plane by swapping groups of cargo which fall into separable regions.

The most relevant previous work with respect to balancing are those of Amiouny et al. (1992) and Mathur (1998). Amiouny et al. (1992) investigate loading items along a single dimension so as to meet a pre-specified target *cm*. In the context of the current paper, their problem is equivalent to loading containers of different width (as well as different weight) along the front to rear axis. They first assume that the *cm* of each container is located at its center point, then later relax this assumption and consider orientation as well. They do not consider the issue of tipping during the loading process. The most basic problem we study assumes equal width containers, but also addresses the constraint that planes must not tip over during loading. We also discuss extensions our model to consider off-center container *cm*'s and container orientation. In both cases our approaches are different than those of Amiouny et al. (1992).

The basic algorithm proposed by Amiouny et al. (1992) works as follows. Each container j has a known weight w_j and width l_j. The target location of the *cm* is C. First, the containers are sorted and indexed by density so that $w_1/l_1 \leq w_2/l_2 \leq \ldots \leq w_N/l_N$. Then the following iterative scheme is applied:

Algorithm Balance

1. Set $c_0 = C$ and $M_0 = 0$
2. For j=1 to N Do:

   Set $p_j = p_{j-1} + \dfrac{M_{j-1}}{\sum_{i=j}^{N} w_j}$

   Locate container j as far as possible from p_j respecting previously placed containers.

In the more complex case where the *cm* of each container is not assumed to be in the center, Amiouny et al. (1992) propose orientation be determined by adding an orientation step: Orient the container so that its *cm* is closest to p_j.

Amiouny et al. (1992) also test a *2-interchange* improvement heuristic in which adjacent containers are swapped if the swap reduces the distance between the *cm* and target. Swapping continues until a local minimum occurs. Amiouny et al. (1992) found that 2-

interchange performed somewhat better than *balance*, but also required greater computation time.

Mathur (1998) provides an improvement to the balance algorithm of Amiouny et al. (1992). As in the case of Amiouny et al. (1992), Mathur assumes a unimodal sequence of weights. That is, the sequence is such that there exists a position $k \in \{1,2,\ldots,N\}$ such that $w_i \leq w_{i+1}$ for $i=1,2,\ldots,k-1$ and $w_i \geq w_{i+1}$ for $i=k,k+1,,\ldots,N-1$. Given this assumption, the problem can be viewed as partitioning the containers into two sets, those to the left of the heaviest container, and those to the right. Mathur shows how the resulting partitioning problem can be formulated as a subset sum problem which he than solves heuristically.

We propose two approaches for solving the loading problem. First we formulate the problem as a mixed integer programming problem and solve it using CPLEX. We show that CPLEX performs reasonably well for a small number of containers (N) and a small optimality gap. However, with increased N and/or a decreased optimality gap, the MIP becomes difficult to solve in a reasonable amount of time, as the application requires the containers to be loaded as soon as possible after being weighed. Thus, we propose a heuristic that follows the procedure proposed by Mathur, with one notable exception; we solve the subset sum problem using a problem space search heuristic with the differencing algorithm of Karmarkar and Karp (1982) embedded. Computational results illustrate the effectiveness of the heuristic.

2. Static Container Load Balancing

We first address the problem of loading containers of known weight so as to achieve the desired center of mass. We assume that the center of gravity of each container is located at the center of the container. As containers are symmetric with respect to the right-left axis, they can be loaded (oriented) either frontward or backward on the plane. Under the assumption that the *cm* lies in the middle, container orientation is irrelevant. We later discuss how to solve the problem when container orientation decisions are present as well.

Schematically the basic static problem is depicted in the following diagram.

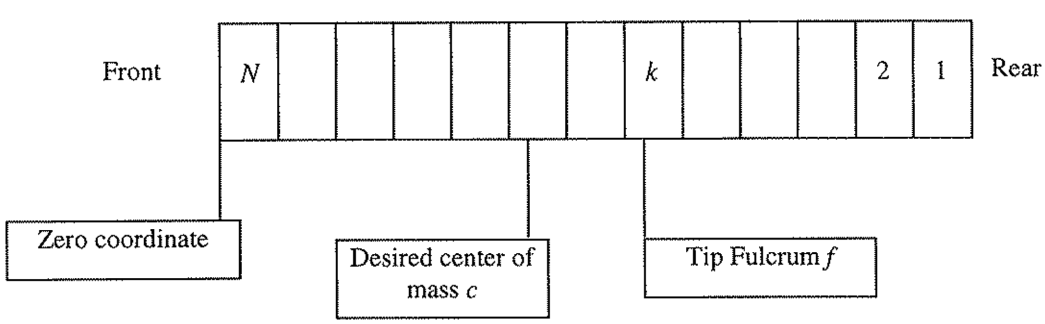


Figure 1. Schematic of the Container Loading Problem

Let the container positions on the plane be indexed by i = 1 to N from right to left (back to front). Let the distance measurement scale be defined so that the width of each container is 1. Let p_i be the location of the center of each container position with respect to the zero coordinate. Thus $p_i = N-i+½$.

Let j = 1, 2, …,N index containers and let w_j be the weight of container j. Let x_{ij}=1 if container j is assigned to position i and 0 otherwise. Let W_0 be the mass, and p_0 be the *cm* of the unloaded plane. Let W be the mass of the fully loaded plane.

Let C be the position of the desired center of mass when the plane is fully loaded. We desire the center of mass of the loaded plane to be as close as possible to C. Thus we seek to:

$$\min \left| C - \frac{1}{W}\left(w_0 p_0 + \sum_{i=1}^{N} p_i \sum_{j=1}^{N} w_j x_{ij}\right) \right|$$

subject to:

$$\sum_{j=1}^{N} x_{ij} = 1 \quad \forall \; i = 1,\ldots,N$$

$$\sum_{i=1}^{N} x_{ij} = 1 \quad \forall \; j = 1,\ldots,N$$

$$x_{ij} \in \{0,1\} \quad \forall \; i,j$$

Thus, this is an assignment problem with a non-linear objective function. To simplify the absolute value in the objective function we introduce positive real auxiliary variables y^+ and y^- as follows. Define the following formulation as MIP:

$$\min \; y^+ + y^-$$

subject to:

$$y^+ \geq C - \frac{1}{W}\left(w_0 p_0 + \sum_{i=1}^{N} p_i \sum_{j=1}^{N} w_j x_{ij}\right)$$

$$y^- \geq \frac{1}{W}\left(w_0 p_0 + \sum_{i=1}^{N} p_i \sum_{j=1}^{N} w_j x_{ij}\right) - C$$

$$\sum_{j=1}^{N} x_{ij} = 1 \quad \forall \; i = 1,\ldots,N$$

$$\sum_{i=1}^{N} x_{ij} = 1 \quad \forall \; j = 1,\ldots,N$$

$$x_{ij} \in \{0,1\} \quad \forall \; i,j \qquad y^+, \; y^- \geq 0$$

As noted earlier, we are concerned with achieving a balanced assignment of containers under the assumption that the plane will not tip while loading.

2.1 Tipping Considerations

Containers are loaded through a door at the front of the plane (position N) and slid to the right to the rear-most unoccupied position. Let the "tip fulcrum" be located at position f, as in Figure 1. This corresponds roughly to the location of the rear wheels so that if, at any point during loading, the location of the current center of mass is greater than (to the right of) f, then the plane will tip over backwards. If a safety margin is desired, f can be decreased. Let positions 1, 2, ..., k index container positions with midpoints to the right of the tip fulcrum.

Assume that the effect of the weight of the unloaded plane can be modeled as a point mass of w_0 located at position p_0. Given an assignment of containers to the first k positions, the moment arm can be expressed as:

$$w_0 p_0 + \sum_{i=1}^{k} p_i \sum_{j=1}^{N} w_j x_{ij}.$$

Let W^k represent the mass of the unloaded plane plus the first k loaded containers. That is:

$$W^k = w_0 + \sum_{i=1}^{k} \sum_{j=1}^{N} w_j x_{ij}.$$

And let W be the mass of the fully loaded plane. Thus

$$W = w_0 + \sum_{j=1}^{N} w_j.$$

After loading the first k containers, the plane can be modeled as a point mass of W^k located at position:

$$\frac{1}{W^k}\left(w_0 p_0 + \sum_{i=1}^{k} p_i \sum_{j=1}^{N} w_j x_{ij}\right).$$

The plane will not tip during loading as long as the following inequality holds:

$$\frac{1}{W^k}\left(w_0 p_0 + \sum_{i=1}^{k} p_i \sum_{j=1}^{N} w_j x_{ij}\right) \le f,$$

or:

$$w_0 p_0 + \sum_{i=1}^{k} p_i \sum_{j=1}^{N} w_j x_{ij} \le f W^k.$$

Substituting in for W^k, the expression can be rewritten as:

$$w_0 p_0 + \sum_{i=1}^{k} p_i \sum_{j=1}^{N} w_j x_{ij} \leq f\left(w_0 + \sum_{i=1}^{k} \sum_{j=1}^{N} w_j x_{ij}\right),$$

or:

$$w_0(p_0 - f) + \sum_{i=1}^{k} (p_i - f) \sum_{j=1}^{N} w_j x_{ij} \leq 0.$$

This constraint dictates that the location of the center of mass when all positions to the right of the tip fulcrum are loaded and all positions to its left are empty must be to the left of the tip fulcrum. Thus, to prevent tipping, this knapsack constraint is added to MIP.

2.2 Generation of Test Problems

We generated a set of 180 test problems by varying three parameters. The first parameter is problem size: we generate problems with N=14 and N=30 containers (which are related to the size of the plane). The second parameter is C_{fact} which determines the desired center of gravity. We can generate upper and lower bounds on feasible centers of gravity (C_L and C_U) as follows. C_L is found by finding the center of gravity when the loading sequence is heaviest to lightest whereas C_U is found from the lightest to heaviest sequence. We then assign the desired center of gravity as:

$C = C_L + C_{fact}(C_U - C_L)$,

where we use C_{fact}= 0.2, 0.5, and 0.8 here.

We assume the weight of the plane (w_0) to be twice the weight of the cargo. The location of the tip fulcrum is assumed to be located at position f = Int($2N/3$), or roughly 2/3 of the way to the rear of the plane.

Our third parameter attempts to control the difficulty of meeting the tipping constraint. Given the container weights for a particular problem instance, we assign the location of the empty plane center of gravity p_0 as follows:

1. Load containers into the positions right of the tip fulcrum starting from the lightest. (This is the configuration least likely to tip the plane).
2. Find location p for the empty plane center of gravity so that the plane just balances when loaded as in 1.
3. Assign $p_0 = P_{fact}*p$ for P_{fact}= 0.2, 0.5, and 0.8

The container weights are generated from a Uniform (0,100) distribution.

We generate 10 instances for each of the 18 factor combinations:

N= 14, 30: C_{fact}= 0.2, 0.5 ,0.8: P_{fact}= 0.2, 0.5 ,0.8;

yielding 180 total test problems.

2.3 Computational Results of MIP with Tipping Considerations

Here we illustrate results of solving the test problems using the MIP with the tipping constraint. The solutions were produced on a 300 Mhz PC with 128 MB of RAM using CPLEX 6.6.

Figure 2. CPLEX Solution Times vs. Optimality Gap

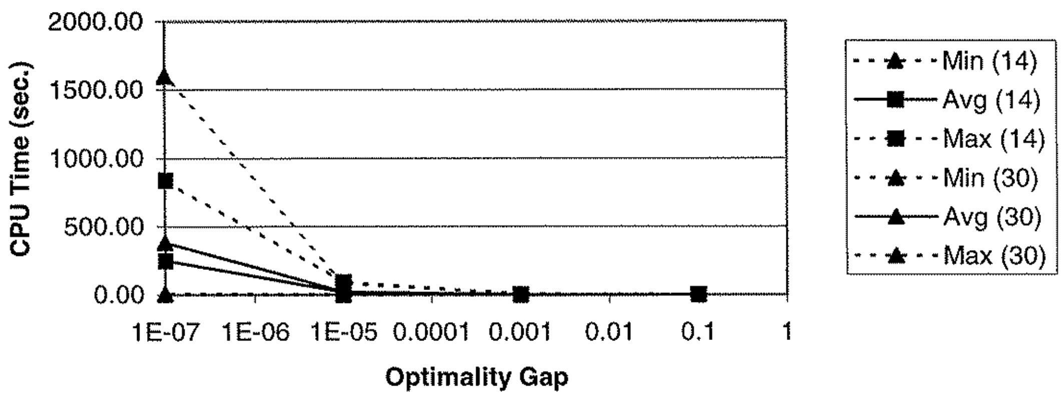


Figure 2 illustrates the minimum, average and maximum CPU time required to solve the 14 and 30 container test problems, respectively, as a function of the optimality gap. As the results show, CPLEX produces fairly good results for larger optimality gaps for both the 14 and 30 container problems. However, as more exact solutions are required, the CPU times grow drastically. For example, with an optimality gap of 0.1, the 14 container problem was solved in 0.24 seconds on average, whereas the 30 container problem was solved in 0.61 seconds. However, as the optimality gap was lowered to 0.0000001, these average solution times increased to 252.1 and 384.8 seconds, respectively. As shown in Figure 2, the maximum solution times for these instances are 841.5 and 1600.4 seconds, respectively.

This application requires that the planes be loaded as soon as possible, after the containers have been weighed, while achieving the best balance. Thus a solution which produces very good solutions quickly is highly desirable. In the following section, we present another approach with very good computational results.

3. Heuristic Approaches to Container Loading Problem

We develop a heuristic based on modified versions of the algorithms described in Amiouny et al. (1992) and Mathur (1998). In these previous works, not only did container weights vary, but the width (along the front to back plane axis) was also assumed to vary. The algorithms we develop will also work with variable container widths, but we test under the assumption of equal widths. Both Amiouny et al. (1992) and Mathur (1998) assume a unimodal or "V-shaped" loading sequence (with lighter containers on the extremes and heavier containers toward the center as discussed in section 1.0). As lighter containers will tend to appear in the rear positions in a "V-shaped" loading sequence, it is reasonable to expect that the tipping constraint will generally be met.

Mathur (1998) formulates the problem of finding the optimal V-shaped sequence as a subset sum problem. Under the V-shaped assumption, once each container is placed either to the left or right of the heaviest container, the loading sequence is determined. Thus the problem becomes one of partitioning the N-1 lightest containers into two groups. Mathur's formulation is:

$$Max \sum_{i=1}^{N-1} A_i X_i$$

$$s.t. \sum_{i=1}^{N-1} A_i X_i \leq (p-\theta)$$

$$X_i \in \{0,1\}$$

where p is the desired center of gravity of the cargo. The reader is referred to Mathur (1998) for determination of the A_i and θ values. Mathur solves the resulting subset sum problems using a modified version of first-fit decreasing. We propose an alternate heuristic to solve the subset sum problem based on the Karmarkar-Karp differencing algorithm.

Storer et al. (1996) have previously developed a problem space search algorithm for the number partitioning problem. Let S be the sum of the A_i coefficients. Then the number partitioning problem is a special case of subset sum in which the right hand side of the first constraint is $S/2$. That is, number partitioning attempts to partition a set of numbers A_i into two sets so that the sum of the each set is as close as possible to $S/2$. Any subset sum problem with target sum T may be recast as a number partitioning problem by adding an additional number R to the set as follows:

Case 1. If $T < S/2$ Set $R=S-2T$
Case 2. If $T > S/2$ Set $R=2T-S$

In case 1, after partitioning into two sets, the desired subset will be in the same set as R (i.e. take the set that contains R and remove R). In case 2, the desired set is the set that does not contain R.

The problem space search algorithm for number partitioning is fully described in Storer et al. (1996). It works by perturbing the data slightly, and applying the well known Karmarkar-Karp differencing algorithm for number partitioning (Karmarkar and Karp, 1982). The algorithm generates 10,000 solutions, each of which is produced by a randomized version of the differencing algorithm. Despite the apparently large (10,000) number of iterations, the algorithm requires very little computation time. Further, it has been shown in Storer et al. (1996) to be remarkably effective on number partitioning problems.

3.1 Computational Results for the Heuristic

The results from our heuristic are shown in Figure 3. Each point in Figure 3 is the distance from the desired center of gravity averaged over the 10 replicates. Note that this measure is equivalent to the optimality gap from MIP as the lower bound in this problem is zero. As shown in the figure, the distances between the desired and actual center of gravity are seen to be remarkably small. The only factor that appears significant is the number of containers. As the number of containers increases, the distance to the desired center of gravity decreases rapidly (note the log scale on Figure 3). This behavior is typical of the subset sum problem and our algorithm. In Storer et al. (1996), it is shown that the number partitioning objective function gets increasingly smaller as N increases. We also checked each solution to see if the tipping constraint was violated. In all 180 problems, the tipping constraint was satisfied.

To put these results in perspective, it must be noted that each of these 180 test problems was solved in under one second, including input and output. (The algorithm was coded in FORTRAN using the same speed machine as CPLEX testing.) Thus, when compared to solving MIP, the 14 container problem produced gaps on the order of .0001 in under one second while MIP required 18.93 seconds on average for gaps on the order of .00001 and 0.54 seconds for gaps of .001. For the 30 container problem, the algorithm produced gaps of .0000001 in under one second while MIP required 384.8 seconds on average for similar gaps.

The remarkable performance of the algorithm raises the question of whether or not such accuracy is necessary. Practically speaking, the weight of containers cannot be measured with infinite precision. To examine the impact of measurement error we undertook a simulation experiment. For each of the 180 loading sequences generated by the algorithm, we added a random variable to each weight, and recomputed the distance to the desired center of gravity. These random measurement errors were assumed normally distributed with mean 0 and standard deviation σ, where σ is varied in our experiment. We generate 1000 simulation replicates for each of the 180 loading sequences, and for each value of σ. The results of this experiment appear in Figure 4. We plot the mean distance over all problems of size 14 and 30, and over all simulation replicates for each value of σ.

Figure 3. Distance from Desired CG

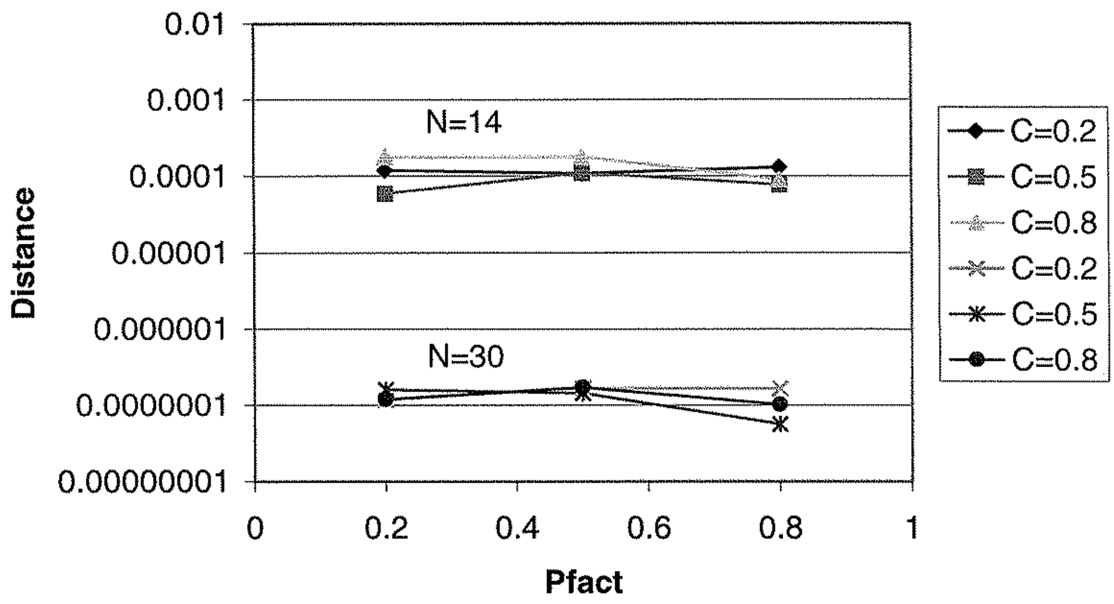
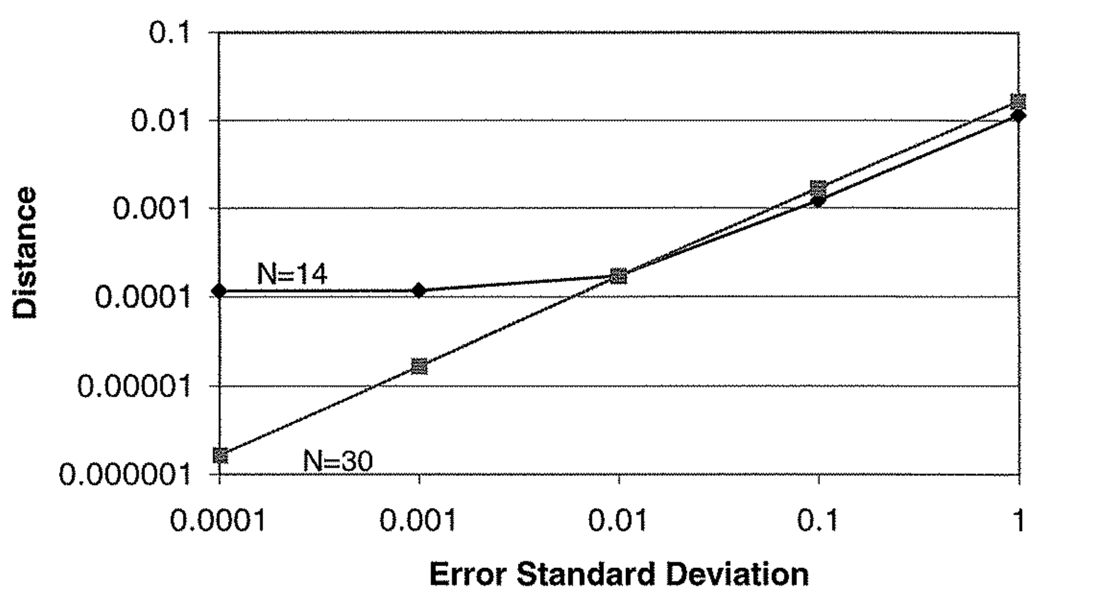


Figure 4. Effect of Measurement Error on Mean Distance from CG



For 14 container problems the results in Figure 4 may be interpreted as follows: if the weight measurement is accurate to 4 digits or less, then the heuristic performance is "within the noise". For 30 container problems, the heuristic results far surpass any reasonable assumptions on measurement error. The conclusion is thus that the heuristic performance is sufficient under reasonable assumptions regarding the accuracy with which the containers can be weighed.

4. Loading Considerations

The heuristic does not explicitly handle the tipping constraints presented in Section 2. It is however reasonable to expect that the V-shaped structure of the solution should yield results that are likely to be feasible to the tipping constraint because higher weighted containers will tend towards the desired center of mass, and thus away from the end of the plane. However, there may be instances in which the solution produced is not feasible. Here, we examine conditions under which it can be guaranteed that our heuristic meets the tipping constraint, and discuss a modification to the presented algorithm to deal with cases when it is not met.

The question we would like to answer here is "When will the algorithm be guaranteed to produce a feasible solution?" We examine this question noting that the solution produced by the algorithm is required to be V-shaped and that the center of mass of the loaded plane is equal to the desired center of mass. This last assumption is justified by the performance of the algorithm. We present two conditions. In the first we assume that the weight of each container is known. This condition can be applied to loads on an individual basis. The second condition assumes only that an upper bound on the weight of each container is known. If this condition holds, all solutions produced by our heuristic will be feasible to the tipping constraint. Both of our conditions assume the following:

1. The loading sequence is unimodal
2. The loaded center of gravity is quite close to the desired center of gravity
3. The following attributes of the airplane are known:
   - the unloaded weight and *cm* of the plane (w_0 and p_0)
   - the desired center of gravity of the loaded plane C
   - the location of the tip fulcrum of the plane f

4.1 Container Weights Known

The basic idea is this: first suppose that very heavy containers must be loaded in the first k positions in order to make the plane tip. Now suppose the plane cannot be balanced with these heavy containers at the rear. Since our algorithm will balance the plane, it will not have these heavy containers in the rear, and thus the plane will not tip.

Our first condition makes the additional assumption that the weight of each container is known. Assume containers are indexed from lightest to heaviest, and let container position k be the position with its cm closest to f such that its $cm > f$.

We first find the lightest container that can occupy position k and still tip the plane. We check the tipping constraint for containers 1,2,..., k to see if the plane tips. If not, we check containers 2,3,..., k +1. We continue until the plane tips. Let d be the key index so that the plane first tips when loaded in the sequence d- k +1,d- k +2,...,d. That is:

$$\textit{Find } \arg\min(d)$$
$$s.t. \left(w_0 p_0 + \sum_{i=d-k+1}^{d} w_i p_i \right) > f \left(w_0 + \sum_{i=d-k+1}^{d} w_i \right)$$

Consider all V-shaped loading sequences with container d in position k. Among these sequences, we find the "key sequence" which is the V-shaped sequence with the least (farthest left) cm. This sequence is:

1 2 3 ... k -1 d d+1 d+2....N d-1 d-2... k

Note that this is the loading sequence so that container 1 is loaded in the back of the plane and k in the front. Let cm_{key} be the cm of the key sequence. If $cm_{key} > C$, then our algorithm will produce a loading sequence that will not tip the plane.

An example will help clarify. Let N=14 and k =5. Suppose the following loading sequences do not tip the plane:

1	2	3	4	5
2	3	4	5	6
3	4	5	6	7
4	5	6	7	8

But the sequence: 5 6 7 8 9, does tip the plane. Then d=9 and the key sequence is:

1 2 3 4 9 10 11 12 13 14 8 7 6 5

Note that determining d and creating the "key sequence" can be performed in $O(N)$ time. Suppose the cm of the plane loaded in this sequence is to the right of C, the desired cm. Since our algorithm produces a balanced loading sequence, this sequence must have less weight in the tail than the key sequence, and thus will not tip the plane. Using this procedure we can very quickly check the condition to see if tipping can be ruled out as a concern.

4.2 Container Weights Unknown

It is also desirable to provide a "no tipping" guarantee that does not rely on knowing the weight of each container. We can apply such a condition to the plane itself, and rule out tipping for any possible load. We derive an alternate guarantee under the same assumptions as the previous section, except that only an upper bound *U* on the weight of any container is known.

Consider the problem of assigning weights in the range [0,*U*] to positions so as to maximize the center of gravity of the plane when only containers to the right of the tip fulcrum are loaded. We maximize this objective subject to constraints:

1. the center of gravity of the fully loaded plane is balanced (= *C*)
2. the weight w_i of all containers obeys $0 \leq w_i \leq U$
3. the loading sequence is unimodal

This problem can be formulated as a linear program as follows:

$$Max \quad \sum_{i=1}^{k} w_i p_i$$

$$s.t. \quad \sum_{i=1}^{N} w_i p_i = CW - w_0 p_0$$

$$w_i \quad form \quad a \quad "V-shaped" \quad sequence$$

$$0 \leq w_i \leq U$$

$$where \quad p_i = N - i + 1/2$$

This LP problem turns out to have a solution that can be found by inspection. Let *CCG* be the center of mass of the containers alone which achieves the overall desired plane *cm* C. That is *CCG* is determined so that the following equation holds:

$$w_0 p_0 + CCG \sum_{i=1}^{N} w_i = C$$

Thus if the weight of a container left of *CCG* is increased, the overall *cm* moves left and visa versa. Let *h* be the index of the first container position with *cm* < CCG (i.e just left of *CCG*). Then the LP solution can be described in two cases:

Case 1: $CCG \geq N/2$. In this case we assign weights to container positions 1 to *h*-1 (i.e. to the right of *CCG*) equal to *U*. We then set the weights of container positions *h* to *N* (i.e. to the left of *CCG*) equal to a constant *Y* so that the plane balances. In this case, the positions at the back of the plane are occupied by containers at the upper bound weight *U* (clearly the worst possible case for tipping). We can easily plug these values into the "no tipping" constraint to see if the condition holds.

Case 2: $CCG < N/2$. In this case we assign weights to container positions h to N (i.e. to the left of CCG) equal to U. Then we set the weights of container positions 1 to h-1 (i.e. to the right of CCG) equal to a constant X where X is the largest value that obeys the balance and upper bound constraints (1 and 2 above). We use the balance assumption to calculate the value of X. The value of X that balances the plane can be shown (with a modest amount of algebra using $p_i = N-i+½$) to be:

$$X = \frac{2B - U(N-h+1)^2}{(2N-h+1)(N-h+2)}$$

where $B = CW - w_0 p_0$

The derived values of the weights of each container (U and X) can then be plugged into the "no tipping" constraint and reduced. The result is that the plane will not tip if the following condition is satisfied:

$$X\left(N - \frac{k}{2}\right)(k+1) \leq fW^k - w_0 p_0$$

where k is the index of the first container position with $cm > f$

Finally we note that our algorithm can be easily modified in the case that it does produce a solution that tips the plane. We simply assign light containers to the first k positions so that the plane does not tip. We include the effects of these containers in the w_0 and p_0, and apply the algorithm to the remaining containers and positions. Thus, in a worst case scenario, the algorithm could be re-solved k times, where k is the number of containers with centers of mass to the right of f.

5. Extensions: Container Orientation

Suppose that each container can be oriented either "backwards or forwards", and that the location of the cm of each container with respect to the "front-back" axis can be measured. The container orientation will clearly effect the center of mass of the loaded plane. As opposed to the solution procedure proposed by Amiouny et al. (1992), we propose to use orientation decisions as a fine tuning mechanism once containers have been assigned to positions. Thus in cases where orientation is an issue, we propose a 2-phase algorithm. We will then show how orientation can be used to balance the cm with respect to the "right-left" plane axis as well as the "front-back" axis.

For the purposes of illustration assume that each container is loaded so that its heavier side is to the right.

Let p_i be the position of the *cm* (with respect to the zero point) of the container located in position i when the container is arranged with its heavy side to the right. Let A_0 be the moment arm of this initial configuration:

$$A_0 = \sum_{i=1}^{N} w_i p_i$$

Then we seek to swap some containers so that after swapping:

$$\sum_{i=1}^{N} w_i p_i = cW$$

As the original configuration under the assumption of a centered *cm* is well balanced, we assume that when all containers are oriented with their heaviest side to the right that the *cm* is to the right of c (i.e. $A_0 > cW$). Let d_i be the change (distance to the left) in position of the cm when container i is swapped. Let S represent the set of swapped containers. We seek to select a set of containers S to swap so that the plane is balanced:

$$\sum_{i \in S} w_i d_i = A_0 - c$$

If we let $a_i = w_i d_i$, and $T = A_0 - cW$, the result is a subset sum problem: find set S so that

$$\sum_{i \in S} a_i =$$

This problem can be very well solved with the same number partitioning heuristic used previously.

6. Conclusions and Directions for Future Research

In this paper, we examine a container loading problem with tipping considerations. A solution assigns containers (equally sized but with different weights) to positions on an airplane for subsequent transport. Ideally, the *cm* is located such that the plane is balanced when flying, thereby not requiring additional operating costs to compensate for the imbalance. We also consider constraints that prevent certain sequences as they may tip the plane during the loading process.

Previous research (Mathur, 1998) illustrates that this problem can be solved as a subset problem. We solve the subset sum problem using a problem space search heuristic with the differencing algorithm of Karmarkar and Karp (1982) embedded. Testing shows that the algorithm provides very good solutions with very little computational effort. This is critical because in overnight shipping applications, it is important to load the containers immediately after weighing. The results are also compared to solving a mixed-integer programming formulation of the problem.

While the presented algorithm does not explicitly handle tipping constraints, its unimodal definition can assume to provide feasible solutions in most applications. We analyze conditions where tipping can occur under the assumption that the weights are known and when only the upper bounds are known. Finally, a modification to the heuristic is provided if the tipping constraint is violated.

Future work will look at a dynamic situation in which the containers arrive according to a stochastic process and their weights become known upon arrival. In this situation, a tradeoff exists between balancing the plane and loading it early (before all containers have arrived) in hopes of assuring an on-time departure.

References

Amiouny, S., Bartholdi, J.J., Vande Vate, J., and Zhang, J. 1992. Balanced Loading. *Operations Research*, **40**(2), 238-246.

Cochard, D.D. and Yost, K.A. 1985. Improving utilization of Air Force cargo loading, *Interfaces*, **15**, 53-68.

Karmarkar, N.R.M., and Karp, R.M. 1982. The differencing method for set partitioning. Report No. UCB/CSD 82/113, Computer Science Division, University of California, Berkeley.

Martin-Vega, L.A. 1985. Aircraft load planning and the computer: description and review. *Computers and Industrial Engineering*, **4**, 357-369.

Mathur, K., 1998. An integer-programming-based heuristic for the balanced loading problem. *Operations Research Letters*, **22**, 19-25.

Storer, R.H., Flanders, S.W., and Wu, S.D., 1996. Problem space search for number partitioning. *Annals of Operations Research*. **10**, 465-487.