Approximating Human Packing for Shipping Container Estimation: a Charitable Endeavor

2015 MAA PIC Math[1]

Eric Bentley, Christian Bottenfield, Michael Garver, Christopher Lindeman, Namita Matharu, Surya Padinjarekutt, Sylvia Ujwary

Jeffrey Paul Wheeler (Faculty)

Department of Mathematics, the University of Pittsburgh

Pittsburgh, PA 15260, USA

jwheeler@pitt.edu

Abstract

The Pittsburgh charity *Global Links* ships surplus medical supplies overseas and wished to not overspend on shipping containers. A technique for estimating the number of needed shipping containers is explained and a computer program optimizing the number of containers using *Global Links* inventory software is presented. The algorithm involves a stochastic process and includes a machine learning component.

[1]Funding for this project is provided by NSF grant **DMS-1345499** through the **MAA *Preparation for Industrial Careers in Mathematical Sciences Program*** **(PIC Math)**, `www.maa.org/picmath`.

1 Global Links

Global Links, founded in Pittsburgh in 1989, is a medical relief and development organization devoted to improving health in resource poor communities, primarily in Latin America and the Caribbean. With this regional focus, Global Links can have an in-depth focus knowledge of the health conditions in the countries it serves. Global Links is unique in their focus of obtaining surplus hospital supplies, other organizations exist which obtain surplus materials from manufacturers

The US health care system produces a large amount of medical surplus which, without the efforts of Global Links, would be destined to fill landfills. Global Links creates a "virtuous circle that converts an environmental burden to a life-saving purpose" (GlobalLinks.org). The Global Links staff makes yearly assessment visits to its nine program countries to ensure that donations of medical assistance are appropriate and useful.

The Global Links organization relies on volunteers to accomplish labor-intensive sorting, preparation, and packing of recovered materials. A local transport company allows them to use available space on their trucks returning to Pittsburgh to bring donations from hospitals outside of the Pittsburgh region. This allows for a low-cost and green shipping option.

2 Problem

2.1 Statement

The surplus supplies from hospitals that are shipped to regions of interest are first packed into cardboard boxes. The boxes are then separated into clinics. For example, Clinic A would consist of a list of particular supplies and a certain amount of each item, Clinic B would contains a list of different quantities of perhaps different supplies, etc. Each clinic is then packed as one contiguous space in the container. Because each shipment costs thousands of dollars, Global Links has asked to know how many shipping containers they will need for each shipment in order to minimize the cost of each shipment.

2.2 Mathematics

For a collection of N items $\{w_k\}$, where $w_k = (x_k, y_k, z_k, h_k)$, we seek a mapping $\varphi : \{w_k\} \to \mathbb{R}^3$. This vector describes the dimensions of the item, along with a height priority, $1, 2, 3$, which corresponds to where the item is usually placed in the z direction. Light, fragile items may have a height priority of 3, indicating placement near the top of a stack, whereas large, heavy items may have a height priority of 1, indicating that they are used as a base on which to stack other items. More specifically, for a partition representing the clinics $C = \{C_i\}$ of the items $\{w_k\}$, and a subspace of $\mathbb{R}^3$ called S, we seek the mapping $\varphi_{clinic} : C \to S$.

S is given by a rectangular prism with a corner at the origin, and restricted by maximum values of KX_C, Y_C, Z_C where K is the total number of containers needed and (X_C, Y_C, Z_C) are the dimensions of said container.

3 Goal

We seek an algorithm which will take inventory data and pack each item into its assigned subspace of the container. By seeking a consistently uniform solution to the number of containers needed to ship some number of packages, we can then adjust for any differences between our program's estimate and the actual number of containers needed on the shipment. With the addition of a machine learning component, the algorithm's accuracy should improve with time, making the estimates more accurately representative of human packing with each iteration. We can use historical data to begin establishing a fair amount of error with which to prime the error factor our program with.

3.1 Best Fit

It is important to be aware of the fact that we will not be seeking an optimal packing arrangement, as one may not even exist. An optimal packing scenario often involves some non-90 degree rotation, which our program will not account for, however it will attempt to rotate packages in random 90 degree increments, in an attempt to consider more possible packing arrangements. By considering more possibilities, we hope to find a relatively best fitting solution. Although we do not expect perfection from the packers, we anticipate that the our best solution will be most accurately adjusted for using the machine learning tolerance portion of our program.

3.2 Constraints

3.2.1 Space

The dimensions of the shipping container will entered by the user. This enables the program to have a wider application than simply for a specific problem.
y_k is the length of an item w_k in the y direction of an item. Y is the width of the shipping container. z_k is the length of an item w_k in the z direction of an item. Z is the height of the shipping container. x_k is the length of an item w_k in the x direction of an item. X is the length of the shipping container.

Where Y_k, Z_k, X_k are the y, z, x-coordinates of where the k-th item is packed in reference to the lower left rear corner of the container.

$Y_k \leq Y - y_k$
$Z_k \leq Z - z_k$
$X_k \leq X - x_k$
$\forall\ k = 1, 2, ..., N$

This simply states that an item cannot be longer than the remaining length of the container in any direction.

Each subspace of the container occupies its own unique space, i.e. $A_i \bigcap A_j = \emptyset$ for $i \neq j$. We seek an upper bound for X, Y, and Z such that the total dimensions needed to ship all items can be expressed as $K(X_C, Y_C, Z_C)$.

3.2.2 Clinic Separation

A clinic C_t is defined as a collection of w_k for some k items $\in \{1, 2, ..., N\}$ where N is the total number of items in the shipment. For the total space filled by a total of M clinics, $\bigcup_{t=1}^{M} C_t$, fills the space exactly. Also, each item belongs to just one clinic $\bigcap_{t \neq v} C_t C_v = \emptyset$.

For each iteration, ψ is a mapping that sends each clinic C_t to a unique subspace S of $\mathbb{R}^3$.

ψ_i: $C_t \rightarrow S$ where $S \subset \mathbb{R}^3$

3.3 Stochastic Rotation

The stochastic rotation component of the algorithm is used to randomly select items to be rotated, thereby considering more possible ways to pack items, mimicking the decisions of human packers.

The purpose of this is to randomly choose if a given item w_k will be rotated laterally by 90 degrees. In other words, the x and y coordinates of the vector that defines the item will be switched from

$$w_k = (X_k, Y_k, Z_k, h_k)$$
$$\text{to}$$
$$\tilde{w_k} = (Y_k, X_k, Z_k, h_k)$$

Given an algorithm that runs i iterations, with a uniform random variable for $\beta_i \in (0, 1)$ and generating $\alpha_{i,k} \in (0, 1)$ for each item, if $\alpha_{i,k} \leq \beta_i \ \forall \ k$, then $\tilde{w_k} = (Y_k, X_k, Z_k, h_k) \ \forall$ items $\tilde{w_k}$. This essentially randomly chooses to rotate the item k for any item whose corresponding $\alpha_{i,k}$ is less than the random β_i generated for the iteration, thereby allowing complete independence in choosing which of the items will be rotated in each iteration. After a preset number of iterations, we leverage the Central Limit Theorem to converge upon the true mean of containers needed.

3.4 Machine Learning

This aspect of our program will accept the user's input of the actual number of containers needed to make the shipment and compare it to the estimate given by the algorithm. We will find an error adjustment factor by the following formula.

$$T_i = \hat{T}_i + N_i(\hat{\xi}_i - 1)^3$$

For the ith shipment, T_i is the number of trucks that was needed, $\hat{T}_i$ is our program's estimate of the number of trucks needed, $\hat{\xi}_i$ is the error adjustment factor, N is the number of items. Below, we calculate the error adjustment factor ξ_{i+1} for the next shipment.

$$\xi_{i+1} = \frac{1}{i}\sum_{r=1}^{i}\hat{\xi}_r$$

3.5 Statistics

Statistical information including the mean, maximum, minimum, and standard deviation will be computed at the end of each iteration's calculation. The final output will display the average number of trucks calculated per shipment in addition to the standard deviation.

4 Main Program Operation

Before inputting data into the program, the data needs to prepared. Firstly, object dimensions must be assigned beforehand. It is necessary to have the dimensions of specific items in order to increase efficiency of our packing solution.

A user tolerance must also be assigned as well as clinic information. When all of this information is assigned, the data can be entered into the program.

The program will first check for input data reasonableness. If the tolerance levels are out of range, there is a flash warning. If the tolerance levels are within the range, the program will begin cycling through the clinics. This is to say that the program will calculate the total clinic volume and place the clinics in the shipping container until the clinic volume exceeds the remaining free space of the shipping container. The method of packing was chosen to minimize the total number of containers required to fit all of the items. Thus, we programmed the algorithm to packing items with a "Y-Z-X" priority. Items are first packed at the "back" of the container to fill the width, then height before moving toward the front of the container. This method ensures that each container as being filled as much as is reasonably possible before moving on to the next container.

Within the cycling of the clinics, the program will cycle through the objects. We take into consideration the priority each item's individual placement with the clinics in order to achieve an effective and accurate solution. For example, the higher priority items must be placed on top while those more weight bearing should be placed at lower levels of the container.

The algorithm will simultaneously be computing several metrics to maintain accuracy in our program. The program will calculate the exterior volume efficiency of each packing algorithm, meaning in the last container. It will also calculate the interior volume efficiency of each packing algorithm in all but the last container. The final output will display the average number of trucks calculated in addition to the standard deviation.

4.1 Algorithm

Before writing the formal pseudo code for the algorithm, we outlined exactly what our algorithm needed to consider in order to achieve an effective and accurate solution. For example, the higher priority items needed to be placed on top while those more weight bearing would be placed at lower levels of the container. Also the packing needed to occur in a Y, Z, X order. This means that items would first be packed along one end of the container, then vertically,and then along the other end of the container.

- We begin by reading data from a spreadsheet
- User inputs now set the needed parameters of the shipment, including specific item data and a calculated error tolerance derived from historical data.
- The program will now index the objects in an initial packing order, based on which clinic C_i it belongs to, its x, y, z dimensions, and its height priority h, as well as assign a uniform random variable $\alpha_{i,k}$ $\forall$ k items.
- The program will now create uniform random variable β_i to compare against each item's uniform random variable $\alpha_{i,k}$.

We first consider the y direction. If $Y_k \leq Y - \sum_1^{k-1} y_r$, then we place the item in the given coordinate. If $Y_k > Y - \sum_1^{k-1} y_r$, consider $Z_k \leq Z - \sum_1^{k-1} z_r$, then $X_k \leq X - \sum_1^{k-1} x_r$. If $X_k > X - \sum_1^{k-1} x_k$, then increment $\hat{T}$ by 1.

Refer to section 6 for a graphical flow chart of the operation of the algorithm.

5 Learning Outcomes

- Creation of Excel Macros
- MatLab Programming
- Machine Learning
- Stochastic Processes
- Heuristic Algorithm Development
- LaTeX

6 Algorithm Flowchart