2006 IEEE International Conference on
Systems, Man, and Cybernetics
October 8-11, 2006, Taipei, Taiwan

A Heuristic Algorithm for the Container Loading Problem with Heterogeneous Boxes

Zhoujing Wang, Kevin W. Li, *Member, IEEE*, Xiaoping Zhang

Abstract— **The container loading problem (CLP) is notoriously known to be NP-hard, an intrinsically difficult problem that is too complex to be solved in polynomial time on a system of serial computers. Heuristic algorithms are often the only viable option to tackle this type of combinatorial optimization problems. This article puts forward a heuristic algorithm based on a tertiary tree model to handle the CLP with heterogeneous rectangular boxes. A dynamic spatial decomposition method is employed to partition the unfilled container space after a group of homogeneous boxes is loaded into the container. This decomposition approach, together with an optimal-fitting sequencing rule and an inner-left-corner -occupying placement rule, permits a holistic filling strategy to pack a container. A comparative study with existing algorithms and an illustrative example demonstrate the efficiency of this heuristic algorithm.**

I. INTRODUCTION

With accelerating globalization of the world economy, rapid growth of container transportation has been witnessed around the globe. To reach its final destination, a container may have to go through a variety of transportation modes such as rail, marine, and trucking. Therefore, it assumes increasing economic significance to utilize the container space in an efficient manner [1, 2]. While the oil price hovers above US$60.00 per barrel as of the first writing of this article in mid-February, 2006, container carriers have additional incentives to make the best use of available container space so that traffic activities can be reduced accordingly, thereby saving costs on fuels.

The CLP usually concerns with arranging rectangular cardboard boxes of cargos in a container in an optimal manner so that the volume use, subject to other practical constraints, of the container is maximized [3]. CLPs can be categorized in different ways [4]. One way is based on a single container or multiple containers. If it is required that a whole batch of goods be loaded completely, multiple containers may be necessitated [5]. On the other hand, if some of the goods may be left behind, a single container is usually involved [6]. Another approach is to differentiate CLPs according to the mix of boxes to be loaded. In this respect, CLPs may vary from a completely homogeneous problem, where all boxes of cargos are identical in terms of dimensions and orientations, to a strongly heterogeneous case, where many different sizes of items are present. The middling situation of relatively few different types of items is often referred to as a weakly heterogeneous CLP [7].

The container loading problem is a well-known NP hard problem [8], which is too complex to be solved exactly in polynomial time. Heuristics are often the only viable option [2, 9, 10]. Recent years have witnessed significant advances in developing various algorithms to handle the CLP [11, 12], and intelligent algorithms are gaining more attentions from researchers, such as genetic algorithms [4, 13], simulated annealing [14], and tabu search algorithms [1], to name a few. Furthermore, researchers start paying more attention to some additional constraints in practice. For instance, Davies and Bischoff [15], Eley [11], and Gehring and Bortfeldt [4] considered the weight distribution of the cargo in the container. Bischoff [16] examines the impact of varying load bearing strength. Loading stability has been accounted for in several occasions such as Bortfeldt and Gehring [13], Bortfeldt et al. [1], and Terno et al. [17]. Other factors like orientation constraints [18] and grouping of boxes [5, 7] in the container are also taken into account.

This paper presents a heuristic algorithm for the CLP with weakly heterogeneous items. This approach employs a tertiary tree structure to represent the container space and a dynamic decomposition method is developed to partition the unfilled space after a block of identical items is loaded. This dynamic decomposition, assisted by a unique sequencing and an inner-corner-occupying placement rule, is designed to search for an optimal partition of the empty space for the next-step packing. When the cargo is loaded, identical boxes are arranged in a larger rectangular block so that they can be loaded into the container in a holistic fashion.

The rest of the paper is structured as follows. Section II briefly describes the problem, followed by an introduction to the tertiary tree model in Section III. The heuristic algorithm is presented in Section IV. Comparative studies with other algorithms and an illustrative example are provided in Section V and VI. The paper concludes with some comments in Section VII.

II. THE PROBLEM STATEMENT

This heuristic is designed to load the cargo packed in

Manuscript received February 21, 2006.
Zhoujing Wang is an associate professor in the Department of Automation, Xiamen University, Xiamen, Fujian 361005, P.R. China (email: wangzj@xmu.edu.cn).
Kevin W. Li is an assistant professor in the Odette School of Business, University of Windsor, Windsor, Ontario N9B 3P4, Canada (Correspondence author, Phone: 519-253-3000 ext 3456, fax: 519-973-7073, e-mail: kwli@uwindsor.ca).
Xiaoping Zhang is with the Department of Automation, Xiamen University, Xiamen, Fujian 361005, P.R. China.

heterogeneous rectangular cardboard boxes into a single standard rectangular container. The dimensions of the rectangular container are given as L (length), W (width), and H (height). Similarly, each box is identified as a cuboid with a length, width, and height of $l_i, w_i,$ and $h_i (i = 1, 2, ..., n)$, respectively. The objective of the algorithm is to determine a loading scheme that maximizes the space utilization of the container.

$$\eta = \max\left\{\frac{\sum l_i \cdot w_i \cdot h_i}{L \cdot W \cdot H}\right\}$$

This algorithm takes a holistic approach to the CLP: subject to the current space volume, homogeneous boxes will first be grouped into a larger block with a rectangular shape and loaded into the container simultaneously. When the algorithm converges, a detailed loading scheme will be generated to specify the optimal sequence and placement of each group.

Similar to that in Ngoi et al. [10], without loss of generality, this algorithm assumes that

1) The cargo is packed in rectangular cardboard boxes only. Other shapes of items are not considered.
2) All boxes are strong and firm enough to bear other boxes placed on top of them. The limited bearing strength constraint [16] is excluded.
3) Boxes are weakly heterogeneous, i.e., there are relatively few sizes and each size has many boxes. This assumption makes the holistic loading approach possible.
4) All boxes are shipped to the same destination so that it is not necessary to prioritize the loading sequence of any box.
5) There is no restriction on the orientation of the box and all boxes may be rotated about x-, y-, and/or z-axis if necessary. For instance, in the case of loading many boxes of shoes or cigarettes into a container, the three dimensions of the boxes can be interchanged without causing any problem.

These assumptions make the CLP more tractable and are applicable to many practical situations in the real world. As this heuristic employs a tertiary tree structure to describe the container space, the basics of the tertiary tree model are furnished next.

III. Basics of the Tertiary Tree Model

In the computer science, a wide variety of data structures are introduced to facilitate the data representation and software development. Trees are one of the most important structures to describe nonlinear information [19]. A tree is often employed to represent hierarchical information, which is defined recursively as a non-empty finite set of nodes,

$$T = \{r\} \cup T_1 \cup T_2 \cup ... \cup T_m$$

where node r is designated to be the root of the tree, and the other nodes are partitioned into subsets $T_1, T_2, ..., T_m$, $m \geq 0$ and each of these subsets is also a tree [19]. Tertiary trees are a special tree structure where either the node set is empty or the set consists of a root, r, and exactly three distinct tertiary trees. In a tertiary tree, a node has either none or three nodes below it, often referred to as children nodes.

Consider the situation that a box or a block of homogeneous boxes forming a large rectangular parallelepiped is placed in the inner left corner of a container as shown in Fig. 1. When the first box or block G(1) is loaded into the container and placed in the corner (labeled as Vol), the unfilled space is partitioned into three subspaces, Vol(1), Vol(2), and Vol(3). Similarly, if the next box or block G(2) is put in Vol(1), the space Vol(1) will be partitioned into three parts accordingly. This partition process continues until all spaces are occupied (it is possible that a space is too small to accommodate any available box, in this case, this unused space is discarded and treated as if it were occupied) or all boxes are loaded into the container.

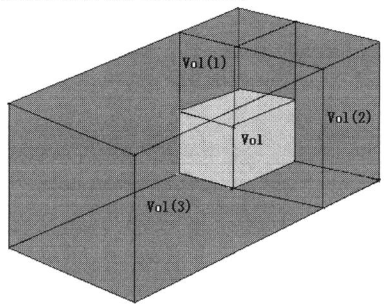

Fig. 1. Partition of the container space when a box/block is loaded.

Given this observation, it is natural to employ a tertiary tree structure to represent the progressive loading of the cargo. Let each box/block be a node in a tertiary tree, the three partitioned subspaces due to the loading of this box/block can be treated as the children nodes of this node. Given that the container and each box/block have certain volumes, the recursive partition process stops after a finite number of steps. Therefore, the resulting nodes are finite and the overall structure is indeed a tertiary tree. An illustration of the tertiary tree is given in Fig. 2.

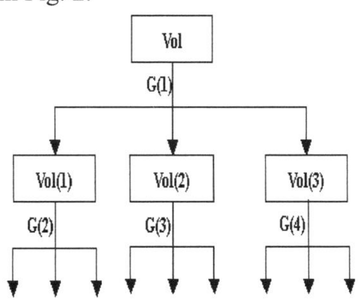

Fig. 2. A tertiary tree model for the space partition.

Next, our heuristic algorithm is formulated based on this tertiary tree structure.

IV. A Heuristic Algorithm

In contrast to other heuristic algorithms in the current literature, where the container space is usually partitioned in a pre-determined manner, our proposed algorithm decomposes the container space based on the tertiary tree model as shown in Section III in a dynamic fashion.

The process starts from the initial full space of the container, which is set as the current space and corresponds to the root of the tertiary tree. According to a unique *optimal-fitting* sequencing rule proposed in this paper, a group of homogeneous boxes that is to be loaded and fits the current space best will be chosen first (See the *optimal-fitting* sequencing rule below for details). A corner-occupying placement rule is adopted to put the cargo in the inner-left corner of the current space. After a block of boxes is loaded, the remaining empty volume in the current space can be divided into three mutually exclusive subspaces, corresponding to the left (*L*), middle (*M*), and right (*R*) children nodes of the root in the tertiary tree. Each of the three subspaces (children nodes) is then set to be the current space sequentially from the left to the middle and, then, to the right node, and the same decomposition procedure is repeated for each new current space until no unused space is available in the container or all unused spaces are too small to fit any box.

One way to partition the remaining space is as shown in Fig. 1. However, there exist several other possible partitions. For instance, in Fig. 3, the initial current space is uniquely determined by its two space diagonal vertices, *A* and *N*, thus denoted by $S(A,N)$. Using the similar space notation, the partition given in Fig. 1 corresponds to $\{S(A,I),\ S(B,M),\ S(C,N)\}$. Other five possible partitions are given as $\{S(F, M), S(A, K), S(C, N)\}$, $\{S(F, M), S(E, N), S(A, J)\}$, $\{S(E, G), S(F, N), S(A, J)\}$, $\{S(E, G), S(A, O), S(B, N)\}$, and $\{S(A, I), S(C, G), S(B, N)\}$ (See Fig. 3 for details).

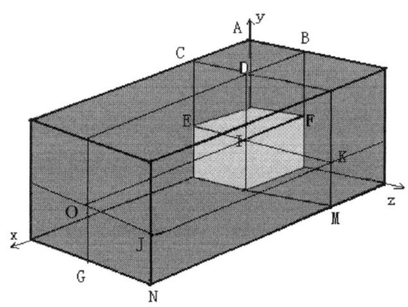

Fig. 3. Different decomposition scenarios for the remaining empty space.

It is obvious that different spatial decomposition scenarios lead to distinct searching paths, which directly affect the efficiency and quality of the solution. The essence of the proposed spatial decomposition approach is to minimize the size of most likely unusable space(s) and maximize the size of most likely usable partitioned space(s). When the cargo is placed in the current space, it follows the sequence of first from the left to the right (along the *z*-axis in Fig. 3) and, then from the bottom to the top (along the *y*-axis in Fig. 3) and, last from the back to the front (along the *x*-axis in Fig. 3). For the resulting three partitioned spaces, the order to set each of them as the new current space follows the same mechanism and this sequential information is encoded in the tertiary tree where the left child node is set first and, then the middle, and the right the last.

To facilitate the formulation of the spatial decomposition strategy, let *L, W,* and *H* denote the length, width, and height of the initial current space $S(A,N)$, respectively, and l_0, w_0, and h_0 represent the length, width, and height of the currently loaded cargo, which is either a single box or a rectangular block of homogeneous boxes. Furthermore, for all boxes to be loaded, assume that $B_{\min}$ is the minimum of any of the three dimensions and $V_{\min}$ is the minimum volume. Given the aforesaid notation, the height of the space $S(A,K)$ is $H_m = H - h_0$ with a volume of $V_m = l_0W(H-h_0)$. Similarly, the space $S(C,N)$ has a length of $L_r = L - l_0$ and a volume of $V_r = (L-l_0)WH$. Now, our dynamic spatial decomposition can be described as the following four rules.

1) If ($H_m \geq B_{\min}$ and $V_m \geq V_{\min}$) and ($L_r \geq B_{\min}$ and $V_r \geq V_{\min}$), the remaining empty space is decomposed into $\{S(F, M), S(A, K), S(C, N)\}$;
2) If ($H_m \geq B_{\min}$ and $V_m \geq V_{\min}$) and ($L_r < B_{\min}$ or $V_r < V_{\min}$), the remaining empty space is decomposed into $\{S(E, G), S(F, N), S(A, J)\}$;
3) If ($H_m < B_{\min}$ or $V_m < V_{\min}$) and ($L_r \geq B_{\min}$ and $V_r \geq V_{\min}$), the remaining empty space is decomposed into $\{S(A, I), S(B, M), S(C, N)\}$;
4) Otherwise, the remaining empty space is decomposed into $\{S(A, I), S(C, G), S(B, N)\}$;

After the first block of boxes is loaded as shown in Fig. 3, rule 1) indicates that space $S(C, N)$ is most likely usable given the two conditions are satisfied, so this partition provides the maximum possible volume for this space. As per the order of loading the cargo into the container, space $S(F, M)$ is most likely unusable and, hence, is allocated the minimum possible volume. While space $S(A, K)$ may or may not be usable, likely depending on the length l_0 as the other two dimensions, *W* and H_m, are both greater than or equal to $B_{\min}$ and $V_m \geq V_{\min}$. By comparing this partition with other five possibilities, we can verify that this is the best decomposition in terms of maximizing the utilization of the current space. Similarly, for the other three rules, each provides the best available partition of the unfilled space for the respective given conditions.

After the unfilled space is decomposed, the next question is to find out the loading sequence of remaining boxes. A variety of sequencing rules has been proposed [9, 10]. The rank of boxes may be determined in a descending order according to their volumes, base areas, the longest of their three dimensions, or aggregate volumes of unloaded homogeneous boxes. This paper proposes a unique *optimal-fitting* rule to rank boxes, which integrates the aggregate volume of homogeneous boxes and the orientation of the boxes together to determine a group of boxes that make the best use of the current space.

To describe the *optimal-fitting* sequencing rule, assume that the cargo to be loaded is packed into boxes with n distinct specifications, $\{I_1, I_2, ..., I_n\}$, and each type I_i has Q_i identical boxes and the corresponding length, width, and height of each box of type I_i are l_i, w_i, and h_i, respectively, $i = 1, 2, \ldots, n$. When a box is placed in a container, there exist six distinct orientations if the "front" and "back", "top" and "bottom", and "left" and "right" positions are not differentiated. Using notation W-H for the rectangle face consisting of the two sides with the width and height and L-W for the face with the length and width sides (other faces can be described similarly), the six orientations can be expressed as: 1. face W-H towards the container door and face L-W towards the top; 2. face L-W towards the door and face W-H towards the top; 3. face L-H towards the door and face L-W towards the top; 4. face L-W towards the door and face L-H towards the top; 5. face W-H towards the door and face L-H towards the top; 6. L-H towards the door and W-H towards the top. For the current space (initially, the full container space), the optimal-fitting sequencing rule selects the cargo according to the following procedure.

1) In the current space, for each specification I_i with the quantity Q_i, calculate the maximum number of boxes that can be loaded as per each aforesaid orientation k, denoted by Q_{ik}, $i = 1, 2, \ldots, n$; $k = 1, 2, \ldots, 6$. It is obvious that $0 \le Q_{ik} \le Q_i$ and the corresponding aggregate volume for each group under each orientation is given as $v(i,k) = (l_i w_i h_i) Q_{ik}$, $i = 1, 2, ..., n$; $k = 1, 2, ..., 6$. In total, $6n$ possible aggregate volumes are obtained. We call the i^* and k^* that maximize $v(i,k)$ as the optimal-fitting specification and orientation, respectively. If such i^* and k^* are unique, then $Q_{i^*k^*}$ boxes of I_{i^*} are chosen and grouped together as a large cuboid for loading into the current space. Otherwise, go to 2).
2) For all i^* and k^*'s that maximize $v(i,k)$, calculate the aggregate length along the x-axis, height along the y-axis, and width along the z-axis (see Fig. 3) of the large cuboid by stacking the identical boxes together. The block with the minimal length (x) is selected first, followed by that with the minimal height (y), and that with the minimal width (z). In the case that all the three dimensions are the same for all optimal blocks, it does not matter which block is chosen first as they are identical. Therefore, a block will be randomly picked up.

This sequencing rule, as a matter of fact, dictates the dynamic spatial decomposition strategy. For instance, after the first group of boxes is loaded as shown in Fig. 3, the second rule ascertains that the usable possibility of $S(F, M)$, $S(A, K)$, and $S(C, N)$ is in an ascending order.

According to the objective of maximizing the container volume utilization and the general container loading practice, our sequencing rule permits a holistic approach to filling the unused space. At the loading time, homogeneous boxes are first grouped together as a large rectangular parallelepiped and, then, loaded to the current space simultaneously (See Fig. 4 for an illustration). For any remaining box with the same dimension, the quantity is updated accordingly.

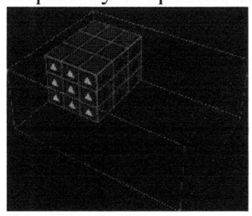

Fig. 4. An illustration of the holistic filling strategy.

The flowchart of the algorithm is as shown in Fig. 5.

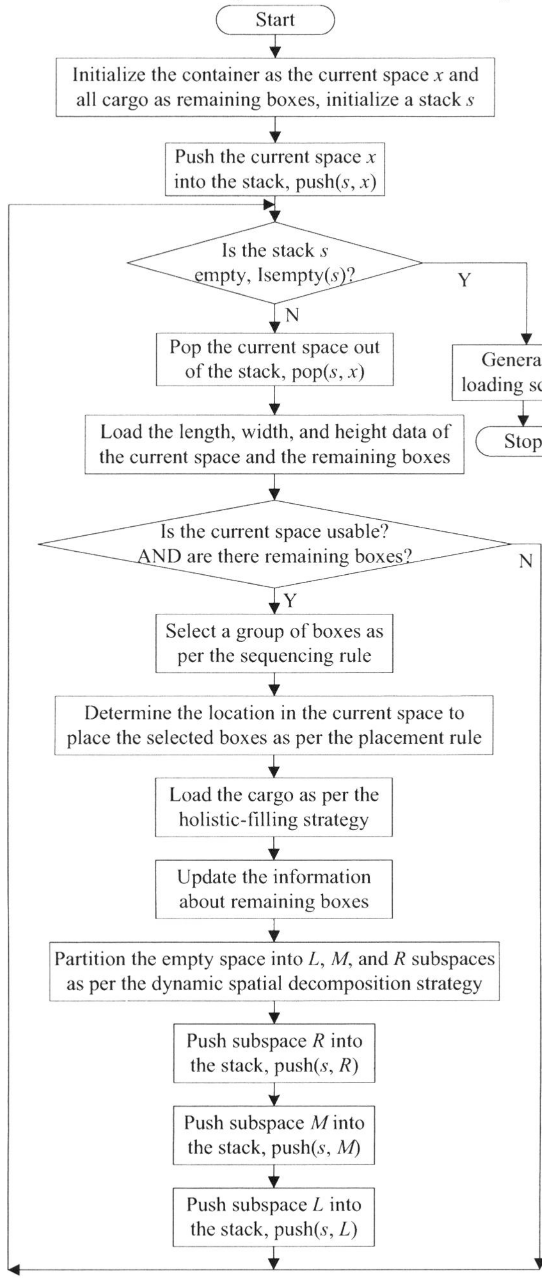

Fig. 5. The flowchart of the heuristic algorithm.

TABLE I
COMPARATIVE RESULT WITH OTHER ALGORITHMS

Problem	Ngoi et al. [10]		Bischoff et al. [21]		Bischoff and Ratcliff [7]		Gehring and Bortfeldt [4]		This algorithm	
	BL[a]	VU[b]	BL	VU	BL	VU	BL	VU	BL	VU
LN01	0	62.5	0	62.5	0	62.5	0	62.5	0	62.5
LN02	54	80.7	23	89.7	35	90.0	39	89.5	35	90.7
LN03	0	53.4	0	53.4	0	53.4	0	53.4	0	53.4
LN04	0	55.0	0	55.0	0	55.0	0	55.0	0	55.0
LN05	0	77.2	0	77.2	0	77.2	0	77.2	0	77.2
LN06	48	88.7	24	89.5	77	83.1	32	91.1	37	92.9
LN07	10	81.8	1	83.9	18	78.7	7	83.3	0	84.7
LN08	0	59.4	0	59.4	0	59.4	0	59.4	0	59.4
LN09	0	61.9	0	61.9	0	61.9	0	61.9	0	61.9
LN10	0	67.3	0	67.3	0	67.3	0	67.3	0	67.3
LN11	0	62.2	0	62.2	0	62.2	0	62.2	0	62.2
LN12	0	78.5	3	76.5	0	78.5	0	78.5	0	78.5
LN13	2	84.1	5	82.3	20	78.1	0	85.6	0	85.6
LN14	0	62.8	0	62.8	0	62.8	0	62.8	0	62.8
LN15	0	59.5	0	59.5	0	59.5	0	59.5	0	59.5
Average		69.0		69.5		68.6		69.9		70.2

[a]BL = No. of boxes left, [b]VU = Volume utilization.

This algorithm is inherently a recursive procedure and the data structure of a stack [19] is employed in the flowchart (as shown in Fig. 5) to transform it to a non-recursive process. Note that the stack structure has the property of first-in-last-out, Fig. 5 guarantees that the sequence of further decomposing the three subspaces starts from the left, to the middle, and then to the right child node.

This algorithm has been programmed by using Delphi 6.0 and OpenGL on a Windows 2000 operating system Pentium PC (The computer program is available upon request). This program allows us to demonstrate this algorithm by using third-party test data and facilitates the comparative studies with other heuristic algorithms in the current literature in the next section.

V. COMPARATIVE STUDIES

The CLP addressed in this article is similar to that in Loh and Nee [20], in which 15 sets of test data are provided. Subsequently, Ngoi et al. [10], Bischoff et al. [21], Bischoff and Ratcliff [7], and Gehring and Bortfeldt [4] tested their algorithms using these benchmark data sets. To demonstrate the efficiency of our algorithm, we use these same test data and the results are shown in Table I along with the results from other algorithms. For the original test data LN01, LN02, …, LN15, see Loh and Nee [20].

The test result speaks for the algorithm and demonstrates its efficiency. The second last column of Table I indicates that only two out of fifteen sets of the test data cannot be completely loaded for our algorithm while the other four algorithms have to leave some boxes out for three to five problems. In the case that other algorithms can completely load the boxes, our algorithm always achieves the same result. For LN07, only our algorithm allows the complete load of all boxes. The last column in Table I shows that our algorithm consistently accomplishes the highest volume utilization whenever none of the five algorithms can load all boxes or only this algorithm loads all. Therefore, the average volume utilization of this algorithm is superior to that of the other four algorithms based on these test data.

VI. AN ILLUSTRATIVE EXAMPLE

This heuristic is also applied to a real-world situation. For the business confidentiality reason, the identity of the firm is omitted. This plastic and rubber producer has contracted to export a batch of its product packed in six different types of carton boxes. The dimensions and quantities of each type of boxes are given in Table II. The objective is to load as many boxes as possible into a standard 20-foot container with three dimensions of $590.5\text{cm} \times 235\text{cm} \times 239.2\text{cm}$. Traditionally, the loading process is carried out by using an empirical approach. According to the empirical calculation, all boxes of type 1 through 5 can be loaded, but for type 6, only 57 of the 129 boxes can be packed into the container. The overall container volume utilization rate is 85.74% as per this loading scheme.

TABLE II
DIMENSIONS AND QUANTITIES OF BOXES

Type	Length (cm)	Width (cm)	Height (cm)	# of boxes
1	51.0	26.0	15.9	47
2	43.0	31.0	17.5	360
3	32.9	22.7	30.6	485
4	51.0	26.0	15.9	69
5	41.0	21.0	24.3	248
6	43.5	31.0	17.5	129

By employing the algorithm proposed in this paper, the whole batch of the cargo can be fully loaded with a total volume utilization of 90.86%. The computer program based on this algorithm can visually demonstrate the loading pattern of different types of boxes as shown in Fig. 6. This windows-based software package allows the user to rotate the loaded container so that he/she may view the loading status in the container from different angles. Furthermore, the user may zoom in and out to make the container and the boxes larger or smaller for a better view. The "animation", "next" and "back" buttons facilitate the user to envision the loading scheme in a progressive manner.

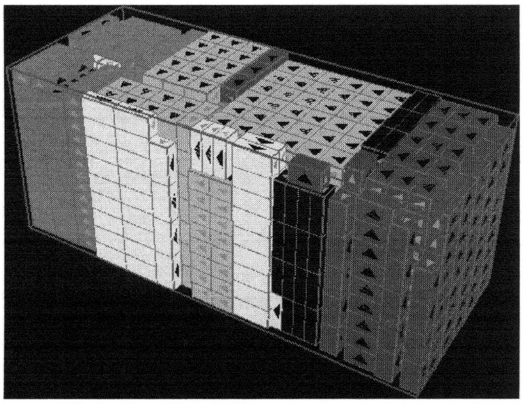

Fig. 6. Visual illustration of the loading scheme.

VII. CONCLUSIONS

A heuristic algorithm is put forward to handle the container loading problem with weakly heterogeneous items packed in rectangular boxes. This algorithm exploits the tertiary tree structure to dynamically decompose the empty space into three subspaces and devises a unique optimal-fitting sequencing rule to rank remaining boxes. Along with the inner-left-corner-occupying placement rule and a holistic filling strategy, this algorithm is able to find a loading scheme that maximizes the container volume utilization. Experiments with the commonly used 15 sets of test data and an illustrative example demonstrate the efficiency of this algorithm and suggest that this approach may be applied in real-world container-loading situations.

This algorithm assumes that there is no restriction on the orientation of boxes, that is, all boxes may be rotated around any of the three axes. However, it is trivial to relax this constraint and fix a particular direction. For instance, if a box must be face up, it may still rotate about the height axis (y-axis in Fig. 3), but not about the x- or z-axis. In this case, only two instead of six orientations are available and our algorithm can be slightly modified to reflect this change.

There are significant issues remaining open. For instance, the algorithm does not consider the distribution of the weight in the container or limited weight bearing strength for any box. In addition, the current algorithm works only for a single container loading problem. These remaining issues warrant further research in this area.

REFERENCES

[1] A. Bortfeldt, H. Gehring, D. Mack, "A parallel tabu search algorithm for solving the container loading problem," *Parallel Computing*, 29(5): 641-662, 2003.

[2] D. Mack, A. Bortfeldt, and H. Gehring, "A parallel hybrid local search algorithm for the container loading problem," *International Transactions in Operational Research*, 11(5): 511-533, 2004.

[3] A. Moura and J.F. Oliveira, "A GRASP approach to the container-loading problem," *IEEE Intelligent Systems*, 20(4): 50-57, 2005.

[4] H. Gehring and A. Bortfeldt, "A genetic algorithm for solving the container loading problem," *International Transactions in Operational Research*, 4(5/6): 401-418, 1997.

[5] S. Takahara, "Loading problem in multiple containers and pallets using strategic search method," In *Modeling Decisions for Artificial Intelligence, Proceedings Lecture Notes in Artificial Intelligence*, Vol. 3558, 448-456, Berlin Heidelberg: Springer-Verlag, 2005.

[6] H. Dyckhoff, "A topology of cutting and packing problems," *European Journal of Operational Research*, 44(2): 145-159, 1990.

[7] E.E. Bischoff and M.S.W. Ratcliff, "Issues in the development of approaches to container loading," *Omega – International Journal of Management Science*, 23(4): 377-390, 1995.

[8] G. Scheithauer, "Algorithm for the container loading problem," In *Operational Research Proceedings 1991*, Berlin: Springer, 445-452, 1992.

[9] J. A. George and D.F. Robinson, "A heuristic for packing boxes into a container," *Computer and Operations Research*, 7(3): 147-156, 1980.

[10] B.K.A. NGoi, M.L. Tay, and E.S. Chua, "Applying spatial representation techniques to the container packing problem," *International Journal of Production Research*, 32(1): 111-123, 1994.

[11] M. Eley, "Solving container loading problems by block arrangement," *European Journal of Operational Research*, 141(2): 393-409, 2002.

[12] D. Pisinger, "Heuristics for the container loading problem," *European Journal of Operational Research*, 141(2): 382-392, 2002.

[13] A. Bortfeldt and H. Gehring, "A hybrid genetic algorithm for the container loading problem," *European Journal of Operational Research*, 131(1): 143-161, 2001.

[14] Z. Jin, K. Ohno, and J. Du, "An efficient approach for the three-dimensional container packing problem with practical constraints," *Asia-Pacific Journal of Operational Research*, 21(3): 279-295, 2004.

[15] A.P. Davies and E.E. Bischoff, "Weight distribution considerations in container loading," *European Journal of Operational Research*, 114(3): 509-527, 1999.

[16] E.E. Bischoff, "Three-dimensional packing of items with limited load bearing strength," *European Journal of Operational Research*, 168(3): 952-966, 2006.

[17] J. Terno, G. Scheithauer, U. Sommerweiss, and J. Riehme, "An efficient approach for the multi-pallet loading problem," *European Journal of Operational Research*, 123(2): 372-381, 2000.

[18] H. Gehring and A. Bortfeldt, "A parallel genetic algorithm for solving the container loading problem," *International Transactions in Operational Research*, 9(4): 497-511, 2002.

[19] B.R. Preiss, *Data Structures and Algorithms with Object-Oriented Design Patterns in C++*, New York: John Wiley & Sons, Inc., 1999.

[20] T.H. Loh and A.Y.C. Nee, "A packing algorithm for hexahedral boxes," In *Proceedings of the Conference of Industrial Automation*, 115-126, Singapore, 1992.

[21] E.E. Bischoff, F. Janetz, and M.S.W. Ratcliff, "Loading pallets with non-identical items," *European Journal of Operational Research*, 84(3): 681-692, 1995.