3D STACKING PALLETIZATION OF MULTI-SIZE BOXES

Jafar Arghavani

A Thesis

in

The Department

of

Mechanical Engineering

Presented in Partial Fulfilment of the Requirements
for the Degree of Master of Applied Science at
Concordia University
Montreal, Quebec, Canada

November 1993

National Library of Canada

Acquisitions and Bibliographic Services Branch

395 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale du Canada

Direction des acquisitions et des services bibliographiques

395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Your file Votre référence

Our file Notre référence

ISBN 0-315-90943-9

Canada

To My Parents
and
To Sophie

ABSTRACT

3D STACKING PALLETIZATION OF MULTI-SIZE BOXES

Jafar Arghavani

The focus of the thesis is on the 3D stacking palletization of multi-size boxes. Boxes of both rectangular and square dimensions are used. The palletization objectives are maximization of the pallet utilization and stability of the stacks, minimization of the work-in-process (WIP) area and of the palletization time. Previous literature on one, two and three-dimensional cutting-stock, packing, and pallet loading problems have been reviewed. Three models are developed to deal with three different types of palletization problems. Model 1 is a mathematical Integer Linear Programming (ILP) model which deals with 3D pallet volumetric optimization. Model 2 provides solutions to 3D stacking palletization, applying interactive 2D palletization and stacking multi-layer procedures. The solutions to the Model 1 and Model 2 problems are obtained using the LINDO (Linear INteractive Discrete Optimizer) software. Model 3 is a random sequence heuristic for 3D stacking palletization, where the number and the availability of non-identical size boxes are unknown and their incoming is random. A program is written in C language to obtain the solution for the latter model. Several examples are illustrated for the three models. The practical limitations are set and reflected in the models. The results obtained from the developed models are compared with those from previous studies.

ACKNOWLEDGEMENTS

I would like to take this opportunity to thank my supervisor Dr George Abdou for his great enthusiasm, his helpful guidance and his supervision throughout the research, and for his support. I also would like to thank Mrs. Sophie Galipeau for her help in the correction of this work

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

NOMENCLATURE

a_{ej}	number of pieces (lengths) of length l_e^* in type "j" strip
a_{k+1}	number of boxes of unit base area
$\hat{a}_i$	zero, if l_i=w_i , for i=N+1,...,2N. $\hat{a}_i$ =1, otherwise
α_{ei}	zero, if l_i=w_i , for i=N+1,.. ,2N, $\forall$e. α_{ei}=1, otherwise
α'_{ei}	zero, if $l_e \neq l_{k+1}$ α'_{ei} =1, otherwise, this represents the unit 1*1.
D_i	number of type "i" box available
D_{jh}	number of optimal box type "j" and height "h"
H	maximum allowable stacking height of the pallet
H_{jr}	remaining height in the column stack "j" on the pallet
h'_h	height of boxes of height type "h", in ascending order for index h=1,2,...,p, such that i.e. h'_1=1, h'_2=2, h'_3=3
h_i	height of box of type "i"
h_k	selected box height for a layer of height HL_j in the column stack "j"
HL_j	height for a layer in column stack "j" on the pallet
ILP	integer linear programming (the decision variables which define the number of boxes of each type are restricted to integer values)
L	length of pallet
LINDO	Linear INteractive Discrete Optimizer software
l_i	length of box of type "i"
l_i*w_i	base area of the box type "i"
l_i*w_i*h_i	volume of the box type "i"
l_e^*	unique length "e" in box length set "s"

N	number of original type of boxes
n_k	number of times each optimum subarea "T" is repeated on the pallet, k=1,2,. .,T
RCH	number of robot motions between the conveyer and holding area
RCP	number of robot motions between the conveyer and pallet
RHP	number of robot motions between the holding area and pallet
RPH	number robot motions between the pallet and holding area
RPS	robotic palletization system
S	box length set containing all the boxes length and width
s	box length set containing unique elements
T	number of different optimum type subareas on the pallet
T_p	total palletization time
T_{ch}	required time for the robot to move between the conveyer and holding are
T_{cp}	required time for the robot to move between the conveyer and pallet
T_{hp}	required time for the robot to move between the holding area and pallet
T_{ph}	required time for the robot to move between the pallet and holding area
U	pallet total volume utilizations
UA	pallet area utilization
UH	pallet height utilization
W	width of pallet
WIP	work-in-process area
w_i	width of box of type "i"
X_i	number of boxes of type "i"

X_{ij}	number of boxes of type "i" in the subarea "j" on the pallet
X_{2N+1}	number of box type of unit area, $l_{2N+1}=w_{2N+1}=1$
y_j	number of strips of type "j"
Z_j	number of optimum subarea "j" on the pallet
Z_{jh}	number of boxes of optimum base type "j" and height "h"

CHAPTER 1 INTRODUCTION

The palletization problem involves interlocking boxes with similar and dissimilar dimensions onto the pallet. Palletization usually takes place in manufacturing environments and distribution centres to ease the transportation of boxes containing all kinds of goods. A common problem experienced by manufacturers and distribution centres is the 'optimal' utilization of the pallet loads. Utilization means how well the loaded boxes use the space in the specified dimensions of length, width, and height of the pallet. Thus, more boxes could be loaded on the pallet the less number of times trips are required to transport a given volume of product between two locations.

Manual palletization is the most common way of loading boxes onto the pallet in small manufacturers and private businesses. It is used for light loads, and flexible environments. These environments are characterized by nonstandard boxes, high flexibility of box sizes, for which their locations on the pallet are not defined and/or they have a random incoming sequence (Penington and Tanchoco, 1988). Here, the operator uses his/her own experience to interlock the boxes of similar and non-similar dimensions onto the pallet. The manual palletization, although time consuming, produces close to optimal results. Therefore, since the stability criterion is an important issue in the palletization, it can be well achieved by the operator's own logic.

The most advanced form of palletization is that which is robotics. Advancement of recent computer and engineering technology has contributed to robotics palletization. Robotics palletization has medium speed and applies to a flexible environment as well. Computers, computer software, vision systems, robots and conveyers are used in robotics palletization. In this case, several programs must be written to control and synchronize the motion of each individual machine to load the boxes onto the pallet. Furthermore, there must be logic programs specially written for the location of the boxes on the pallet.

No complexity involves when loading a pallet with identical boxes however, many different criteria have to be considered in the palletization with multiple size boxes, such as:

- random sequence arrival of boxes
- palletization time
- types of goods carried in each box
- grouping different types of boxes
- loading priority
- unloading priority
- loading and unloading time

Generally, both manufacturers and distribution centres share the following common objectives to obtain solutions to the palletization problem:

- maximization of the pallet volume utilization
- maximization of the value of different types of boxes to be loaded on the pallet
- reduction in the pallet loading time

• reduction in the pallet unloading time

• loading and unloading priorities with respect to different demands

• maximization of the stability of the load

• reduction in the WIP area

Different approaches, including exact, and heuristics, can be used to deal with the palletization problem; in all approaches certain objective functions, similar to those mentioned above, must be developed. Also, certain constraints, with some specifications, must be developed to satisfy the objective functions, for which the objective functions can be maximized/minimized. These constraints are subjected to upper and lower bounds and can be defined as:

• pallet constraints,

• box availability constraints,

• stability constraints,

• sub-area constraints,

• work-in-process constraints,

• etc.

Manufacturers have been aware of the importance of the palletization problem for some time. Therefore, researchers have given considerable attention to this problem. The previous works have been primarily focused on the application of various techniques to deal with the mathematical aspects of the palletization problem. However, since these mathematical aspects do not always give feasible results, heuristic should be dealt with. Furthermore, few researchers also have used mathematical and heuristic approaches to deal with the physical aspects of implementing a pallet loading system.

In most cases, the pallet dimensions are fixed, and the palletization problem arises from two points of view: the manufacturer's problem and the distribution centre's problem. The palletization usually occurs in large, busy and fully-automated centres. Usually, the pallet loading takes place at the end of the production line, and the manufacturers deal with a random pattern of incoming sequences of boxes of different dimensions. One may assume that some manufacturers pack their products into identical boxes, so the palletization process at the end of production line is done all with identical boxes. However, for a manufacturer with multiple production lines, boxes of different dimensions may randomly arrive on the final conveyor. On the other hand, distribution centres usually deal not only with a known numbers and types of boxes, but also with the random arrival of a sequence of boxes of different dimensions carrying different types of goods. Therefore, the palletization in both cases is not an easy task. Furthermore, palletization is done using certain patterns that are discussed next.

In general, cutting-stock, packing, container loading and palletization are done in either orthogonal or non-orthogonal patterns. In terms of cutting-stock, rectangular units can be cut into smaller pieces in either orthogonal or non-orthogonal patterns. Whereas, in terms of palletization, pallets can be loaded with number of boxes in either ways. In orthogonal patterns small items, such as boxes, are arranged parallel to the large object's edge such as pallets, whereas in non-orthogonal patterns the angle at which the boxes are loaded on the pallet is optional. In the orthogonal pattern either a guillotine or nested pattern cut can be employed. Therefore, boxes can be loaded on the pallet with guillotine-cut pattern or nested pattern. The guillotine-cut patterns are continuous from one end of the object to the other end (figure 1). However, interrupted cuts are allowed at any point in the nested pattern (figure 2), H. Dyckhoff and U. Finke (1992).

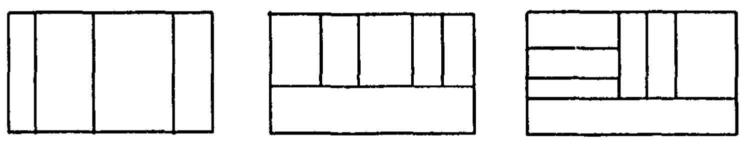


Figure 1. Orthogonal Guillotine-cut pattern

single-stage two-stage three-stage

The guillotine-cut pattern can be divided in different stages such as single-stage, two-stage, three-stage, etc. This may depend on the design criteria (with respect to the types of boxes available), some loading and unloading priority, or the manufacturer and distributor requirements. The solutions to the problem with guillotine-cut patterns are carried out more easily than those of the nested pattern.

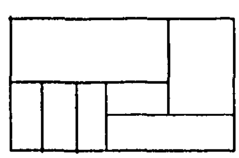


Figure 2. Orthogonal Nested pattern

Palletization problems are complicated when dealing with three-dimensional palletization with random arrival sequence of the boxes. Mathematical and heuristic models are two common approaches dealing with the solutions of palletization problems. The mathematical approach in palletization is applicable and is more likely suitable for the material handling system in a warehouse, or a distribution centre, where all the merchandise is in stock. In this case, availability of boxes of all types is certain, the dimensions of all boxes and pallets are known and pre-specified, and the number of boxes required at any time is known.

Heuristic approaches are required to the solution of the palletization problem if the availability of each type of box is in question, such as in the situation where the boxes on the conveyer are arriving in a random sequence. For such a case, there is no known mathematical approach. Heuristic approaches, with some practical assumptions, are considered to deal with complex optimization problems since no exact mathematical method for solving the three-dimensional palletization problem, with random arrival sequence of boxes, has been developed which yields an optimal solution.

In this research, the 3D type stacking palletization problem is studied. Three models are proposed. Some applications are applied to illustrate the efficiency of the proposed models.

CHAPTER 2 LITERATURE REVIEW

Since the early 1960's, the pallet-loading problem has prompted researchers to look into many criteria and to develop many procedures for the solution of palletization problems. Such a problem arises in a variety of situations including cutting-stock problems, packing, container loading and placement problems. Many researchers have focused on the modelling and the solution of the problem in two, two and half, and three-dimensional problems. Several approaches, including graph-theoretic, tree-search, exact, and heuristics, have been used. Most research has focused primarily on three approaches to obtain optimum solution to palletization problem: mathematical programming, dynamic programming and heuristics.

The characteristics of one, two, 2½, three, and four-dimensional problems can be defined as follows:

• The one-dimensional problem can be defined such that objects of pre-determined lengths are chopped into smaller pieces. The two-dimensional cutting-stock or packing problems are variant. If the required number of pieces of each type is unbounded, and the layouts are restricted to those obtained by guillotine cuts, the problem has to be categorized into the dynamic programming.

• In two-dimensional palletization problems two cases can be considered since the problem deals with surface partitioning: the single layer pallet loading with objects of same dimensions, or objects of different base dimensions with their heights being similar.

• In the 2½-dimensional palletization problem multi-layer pallet loading is encountered. The objects are of the same dimensions, or have different base dimensions with similar heights.

• The three-dimensional pallet loading problem deals with loading objects of different size on the pallet, where the objects have different lengths, widths and heights.

• The four-dimensional (conventionally called multi-dimensional) palletization problem deals with loading objects of different dimensions as well as considering the time period required to load the pallet.

Three types of pallet loading usually can be considered for the solution of the palletization problem: Layered palletization, Mixed palletization, and Column-stacked palletization. In the layer-type palletization, boxes are loaded layer by layer onto the pallet and the height of each layer is constant. In mixed type palletization, boxes are loaded onto the pallet as long as there is available space; whereas, in the stacking palletization, pallet is partitioned into smaller areas, namely subarea, and then each subarea is stacked until the pallet stacking height is obtained.

There has been much research concerning two-dimensional layer palletization. Also, recently some have developed algorithms to deal with the solutions to the 2½-dimensional and three-dimensional layer palletization. Most of the work in this area has concentrated on the palletization with boxes of similar dimension, and only a few researchers have studied the palletization problem with boxes of different dimensions, where layer palletization by grouping boxes with respect to their similar height has been worked on However, there is no literature on 3D stacking palletization with multi-size boxes, furthermore, no ILP model has been reported.

There has not been much attention given to the three-dimensional stacking palletization problem with boxes of different dimensions. Most of the literature has dealt with two and 2½-dimensional layer palletization, or loading a container, with either boxes of the same dimension, or boxes of different base dimensions with similar height

The earlier research into the pallet loading problem can be viewed as an extension of work on the one and two-dimensional cutting-stock problems. Gilmore and Gomory (1961-63-65) showed how the one-, two-, and three-dimensional cutting-stock problem can be solved using mathematical programming techniques. They considered that the packing problem is essentially the same as a cutting-stock problem. However, they did not present any algorithm for the three-dimensional problem for which an optimal solution can be obtained within reasonable computer time Only heuristic approaches are proposed for practical packing problems.

Many have considered that the manufacturer's pallet packing problem involves producing and packing products in identical boxes and stacking the boxes on identical pallets. Steudel (1979), Smith and DeCani (1980), Bischoff and Dowsland (1982) and Dowsland (1982) presented procedures to the solutions for this type of problem.

George and Robinson (1980) developed a heuristic approach for packing boxes into a container In their research, the dimensions of the container are known as well as the collection of the rectangular boxes with known dimensions and the number of boxes available of each type. Their objective was to find a suitable position for packing the boxes in the container in such a way that all the boxes can be fitted in. The heuristic algorithm was based on the concept of filling the container layer by layer. The restriction was that the upper layer starts only if the lower layer is completed. However, they defined an empty space to be filled at the later time, if there is no box available to fill the space on the working layer. The problem considered packing groups of identical boxes into more than one container. The problem was solved under the restrictions that the boxes of the same type were to be stacked in proximity and adjacent to each other.

Hodgson (1982) discussed the 3D palletization problem by introducing layer palletization with boxes of similar height, and stacking palletization with pre-specified stacking column height. In the latter case, the pallet was loaded with columns of boxes such that the stacks are no higher than the maximum allowable pallet height, however, no results were reported. He also presented two approaches for reducing the CPU time for the dynamic programming based on the heuristic algorithm.

Brown (1980), "An Improved BL Lower Bound" (BL stands for Bottom-up Left-justified) developed an algorithm which packs small rectangular pieces into a larger rectangular. Completion to that, Tsai et al. (1988) used the Brown's method and developed an Integer

Linear Programming (ILP) model for a two-dimensional palletization problem, for which boxes of various types might be loaded on the pallet base. This model only considered one layer on the pallet, and assumed that all the boxes had the same height Note that in the ILP the decision variables, which define the number of boxes of each type to be loaded on the pallet, are restricted to integer values

Loschau (1989) investigated the stability criterion of the column stacks Since stability is the major concern of stack palletization, Loschau used the critical inclination angle as the criterion for the rigid block, and deviation from the centre of gravity for the flexible bar model. The rigid block was considered as boxes are loaded on the base of the pallet and the flexible bar was when the boxes are loaded on the top of other boxes

H. Gehring et al. (1990) developed a computer-based heuristic model for packing pooled shipment containers. It considered the packing of rectangular boxes of different size in a shipping container of known dimensions. Its objective was to determine positions for placing the boxes in the container such that the waste of space is minimized. For this three-dimensional cutting-stock problem various sub-optimal solutions were generated using computer-based heuristics. Their algorithm consisted of a pre-determined number and dimension of boxes, and the container was filled layer by layer.

Haessler and Talbot (1990) discussed the problem of packing low density products in railcars, truck trailers and tandem truck trailers. In railcars and trailers, the boxes had to be stacked in layers of unit loads. The restriction was the placement such that the unit loads were of rectangular shape and were designed to ship in railcars two high and two across the width of the car. The boxes were grouped in unit size blocks; blocks with nearly identical box dimensions.

Chen et al. (1991) presented a mathematical model to pack non-uniform box sizes onto a pallet The problem was formulated as a zero-one integer programming problem, which included multiple-pallet loading. The mathematical model was a mixed Integer Linear Programming problem. However, their mathematical model of the pallet loading problem could not be solved directly since this algorithm depended solely on the number of pallets "F" used, which was unknown and had to be determined interactively. This formulation was only capable of finding the feasible solutions for only one layer on the pallet which in fact can be considered as a model for the two-dimensional pallet loading problem with different rectangular pieces. The research considered boxes of uniform height.

Dowsland and Dowsland (1992), in their survey, refer to three-dimensional packing problem as: "Much of the published work in this area has been concerned with packing of shipping container, often referred to as the container stuffing problem. The increased combinatorial complexity of this problem over the two-dimensional case means that exact solutions are unlikely to be effective. Models for the two-dimensional case such as that suggested by Beasley could theoretically be extended to include the third dimension. We are not aware of any published results in this area, although Mannchen (1989) has devised a tree-search algorithm which he reports as working successfully for packing of non-identical rectangular pieces in two and three dimensions."

Abdou and Lee (1992) developed an algorithm to deal with the three-dimensional palletization problem where boxes were grouped with respect to their similar height. The algorithm is capable of supporting different pallet sizes and different box types if required. The restrictions in this algorithm were fixed pallet dimensions and the pallet stacking height. The layer palletization was considered and the maximum area of each layer was restricted to the pallet size as well as the area of the lower layer. They obtained better results in terms of pallet utilization, palletization time and the WIP area than the other algorithm mentioned in their literature.

Abdou and Yang (1993), and Yang (1993) developed an algorithm to deal with multi-layer palletization of multi-size boxes for 2½D and 3D problems. They considered both the mathematical and heuristic approaches, and emphasized the maximization of pallet utilization, the minimization of work-in-process area, the minimization of the palletization time, and static loading stability. Yang employed Linear Integer Programming for the 2½D palletization problem, the systematic procedure and the heuristic approach for the 3D Problem. The physical robot palletization system consisting of Gantry robot, vision system, detecting sensors and conveyer system was implemented to test the developed model.

These studies have approached palletization, cutting-stock and packing, problems from different angles and have obtained interesting results. However, there are still many aspects of the palletization problem that have not been thoroughly studied. The lacunae observed from the literature review are as follows:

• most studies discussed palletization with identical boxes,

• studies mostly focussed on 2D problems,

• layer palletization has been the target for loading,

• only few have studied the 3D palletization, using layer pallet loading concept,

• no integer programming model has been reported for 3D problem.

• no 3D stacking palletization with boxes of different dimensions has been found.

The algorithms' limitations and the advantages of the stacking palletization are reviewed in the following sections.

2.1 Limitations of the palletization algorithms

There have been certain limitations in the algorithms developed by researchers to deal with the palletization problem. For instance, boxes were only allowed to rotate about the Z axis. In this case they could only have two orientations. The limitations to multi-layer palletization algorithm have been observed as follows:

• boxes of the same dimensions or similar height are used,

• boxes of multi-size have to be grouped with respect to their height in order to be loaded in a layer on the pallet,

• the height of the entire layer remains constant in the layer palletization,

• an upper layer can not be loaded with boxes unless the lower layer is completely filled,

• boxes on the lower layer can not be accessed without removing the boxes of the upper layers,

• the stochastic process, random sequence arrival, of boxes of different dimensions may not result to a feasible solution in layer palletization where the availability and the number of each type of box is not known.

2.2 The main advantages of stacking palletization

The advantages of stacking palletization can be easily viewed from the limitations mentioned above, and the objectives of pallet loading. In most cases of both pallet loading and container loading, the concept of loading layer by layer with boxes of identical dimensions, or at least similar height, have been considered. Hence, the following highlights some of the advantages of stacking palletization:

• boxes of different dimensions, different lengths, widths and heights can be used,

• the arrival of boxes of different sizes can be in a random sequence,

• the number and the availability of each type of box is not considered,

• any type of boxes can be loaded on any sub-area (small partitioned area) onto the pallet at any time, depending to the algorithm developed,

• pallet stacking is suited for the random sequence arrival of boxes of different dimensions in terms of first come first served, such that the newly arrived boxes can be immediately placed in a sub-area on the pallet,

• boxes can be loaded on top of each other to stack up, depending on the requirement of the algorithm, even where there is a sub-space available on the pallet area, providing the stability criterion is considered,

• stacking the pallet in terms of last-in first-out is best suitable in the case of distributing centres, where there are boxes of different types and there are different market demands, such that the large number of boxes can be accessed in any order,

• stacking is also appropriate where boxes can be grouped and stacked up in terms of the goods they carrying in, their dimensions and in terms of unloading priority requirements,

• each stack may have different locations on the pallet, so that boxes can be easily accessed,

• stacking height is different on each specific column, since boxes are of different dimensions and arrive in a random sequence.

The summary of the literature is shown in the table 2.1 in the next page.

Table 2.1 Summary of Literature Review

Attributes	Authors	Abdou and Yang 1993	Abdou and Lee 1992	Chen 1991	Tsai 1988	Hodgson 1982	Proposed 1993
Dimension	2D			✓	✓		
	2½D	✓	✓			✓	
	3D	✓				✓	✓
Availability	limited	✓	✓	✓			✓
	unlimited	✓			✓	✓	✓
Loading Pattern	nested	✓	✓	✓	✓		✓
	guillotine						✓
Objective	pallet utilization	✓	✓	✓	✓	✓	✓
	stability	✓					
	WIP	✓	✓				✓
	loading speed	✓	✓				✓
Loading Stage	single stage	✓	✓	✓	✓		
	multi-stage	✓					✓
Procedure	mathematical	✓		✓	✓		✓
	heuristic	✓	✓			✓	✓
	mixed	✓					✓
Palletization Method	layer Palletization	✓	✓	✓	✓	✓	
	stacking						✓
	layer & stacking						✓
	volumetric						✓

CHAPTER 3 OBJECTIVES OF THE RESEARCH

The objective of the research is to develop algorithms on stacking palletization, using different types of multi-size boxes to achieve more efficient solutions to the palletization problem than those already existing in the literature. Thus, the intention is to review the research done in the area of palletization and address the unsolved problems. However, it should be mentioned that the rectangular loading problem is known to be NP complete (Girkar et. al. 1992), hence, it is often not possible to obtain an exact optimal solution within a reasonable time.

The emphasis of the research is on the 3D palletization problem, using both mathematical and heuristic approaches, with consideration of some constraints as follows:

• Boxes of multi-size are used, the boxes are allowed to rotate about the Z axis and their rotation about X and Y axes is not considered. Hence, boxes may have two different orientations on the pallet.

• The pallet dimensions and the stacking height are fixed, and pre-determined.

• Stacking palletization is considered.

• No over-hanging is allowed, such that the total area of the boxes on the pallet are less than or equal to the total pallet area. Also the area of the boxes on each subarea (smaller area defined on the pallet) on the pallet must not exceed that area.

• The height in each column stack must be less than or equal the pallet maximum stacking height.

With respect to the assumptions mentioned above four main objectives are considered in the research:

• **Develop Model 1:** A mathematical ILP model for 3D pallet volumetric optimization.

• **Develop Model 2:** A mathematical model for 3D stacking palletization.

• **Develop Model 3:** A random sequence heuristic algorithm for 3D stacking palletization, where the number and the availability of non-identical size boxes are unknown and their incoming is random.

• **Analysis:** Compare the results from proposed models with those generated by the existing algorithms in the earlier literature.

The proposed mathematical Model 1 and Model 2 are suitable for warehouse/distribution centres, where all the information about the numbers and types of boxes are known Model 1 optimizes the pallet volume with boxes of multi-size however, it does not provide a pre-determined location of the boxes on the pallet, and the loading pattern solely depends on the number and types of boxes obtained, which can be either layered or stacked palletization. Model 1 may be used for 3D cutting-stock, warehouse allocation problem and placement. On the other hand, to complement Model 1, Model 2 is developed to provide stacking procedures for which the location of the boxes on the pallet is pre-determined.

To assess the stacking palletization problem in the stochastic process, random sequence arrival, with boxes of different dimensions, Model 3 is developed. This model is best suitable for the manufacturing systems where the palletization takes place at the end of production line with a stochastic process of multi-size boxes.

CHAPTER 4 PERFORMANCE MEASURES

To assess the palletization performance, in the algorithms proposed in the research, some criteria are required. In most of the studies in this area, researchers have been mainly focusing on the pallet utilization as the main objective. This has been mostly because of the requirements of their algorithm and also because they were dealing with boxes of similar dimensions, where the number and availability of the boxes have been known. However, when dealing with different types of multi-size boxes for which the number and types of boxes available are not known in advance, and their arrival is in random sequence, the solution to the problem should not be limited to the pallet utilization, but, some other factors must be considered. In the real world where the physical palletization system is used, the palletization performance could be assessed by: Pallet utilization, Work-In-Process (WIP) area, Palletization Time and the Stability of the stack.

4.1 Pallet Utilization

Pallet utilization U is expressed as the pallet area utilization UA times the pallet stacking height utilization UH, where the pallet area utilization UA is expressed as the sum of areas of boxes which are loaded on each strip/optimum subarea "j" on the pallet divided by the maximum pallet area.

$$UA = \frac{\sum_{i=1}^{2N} \delta_i * l_i * w_i * X_{ij}}{L * W} \quad ; \qquad \forall j \tag{4.1}$$

Pallet height utilization UH is expressed as the sum of the heights of boxes which are loaded on each subarea "j" on the pallet divided by the maximum allowable pallet stacking height.

$$UH = \frac{\sum_{i}^{N} h_i * X_{ij}}{H} \qquad \forall j \tag{4.2}$$

In general, pallet utilization in terms of volume can be expressed by the total volume of the boxes loaded on the pallet divided by the maximum palletization volume.

$$U = \frac{\sum_{i=1}^{2N} \hat{a}_i * l_i * w_i * h_i * X_{ij}}{L*W*H} \qquad \forall j \tag{4.3}$$

where;

$\hat{a}_i=0$, if $l_i = w_i$ for i=N+1,.... 2N; $\hat{a}_i=1$, otherwise,

l_i and w_i are the length and the width of the box type "i" on the pallet respectively,

is the width of the box type "i" on the pallet,

L and W are the pallet length and width respectively,

h_i is the height of the box type "i",

X_{ij} is the number of boxes of height type "i" on subarea "j" on the pallet.

4.2 Work-In-Process (WIP)

Work-in-process is expressed as the waiting area between the conveyer and the pallet for which boxes are waiting to be loaded on the pallet. The size of the WIP is defined as the total area of the boxes held and not loaded on the pallet or the number of boxes held on the WIP. The size of the holding area changes as the number of the boxes in the WIP area changes.

In the robotic palletization system (RPS) boxes are moved from the conveyer, by the robot arm, to the pallet and if the boxes are not of the type to be loaded on the pallet at the time, they are moved to the WIP area either from the conveyer or from the pallet. The number of boxes in the WIP area has a direct impact on the movement of the robot arm and on the palletization time. The increase in the number of boxes in the holding area increases the robot movement between the conveyer and the WIP and between the holding area and the pallet, hence palletization time increases.

4.3 Palletization Time

Palletization time in a physical robotic palletization system (RPS) depends not only on the movement of the robot between the conveyer system and the pallet, the conveyer and the WIP area, the WIP area and the pallet, the pallet and the WIP area, but also on the algorithm of the palletization, interfacing between other systems (i.e. computer, etc.) and the robot arm, the vision system, and the programs used to interface these systems. Since the CPU time of the conveyer system is considerably shorter it can be neglected. Furthermore, since the reaction time for different parts of the robot depends on the commands from the computer, it is difficult to obtain the idle robot time. Hence, in this research the palletization time is concluded only for the robot motion during the palletization process, which undertakes the conveyer speed time, vision time, CPU time and robot reaction time.

The robot motion during palletization process can be summarized as: moving a box from the conveyer to the pallet, moving a box from the conveyer to the holding area, moving a box from the holding area to the pallet, and moving a box from the pallet back to the holding area. The total palletization time required in a physical robotic palletization system is:

$$T = T_{cp} * RCP + T_{ch} * RCH + T_{hp} * RHP + T_{ph} * RPH \qquad (4.4)$$

Where; T_{cp}, T_{ch}, T_{hp} and T_{ph} represent the time required for the robot to move from the conveyer to the pallet, the conveyer to the holding area, the holding area to the pallet, and the pallet to the holding area respectively. RCP, RCH, RHP and RPH represent the number of robot motions between the conveyer and the pallet, the conveyer and the holding area, the holding area and the pallet, and the pallet and the holding area respectively.

4.4 Stability

Stability is an important issue in the palletization problems, especially when dealing with boxes of different dimensions. The stability is measured by an index which indicates the ability of boxes to maintain their positions on the pallet in both static and dynamic situations. In the stationary stack boxes are subjected to the forces caused by other boxes. In addition, boxes are subjected to dynamic forces during transportation.

Very few studies have been reported about the stability criterion in the loading problem. This has been mainly because most of the studies on the palletization involved loading with identical boxes. Loschau (1989) discussed the stability criterion of column stacks, on the basis of an inclination angle, different methods for obtaining the stability criteria were presented. He defined a "rigid block" model for the case where boxes loaded on the base of the pallet, and a "flexible bar" model for boxes which loaded on the top of other boxes and which were the target of vibration during transportation.

Carpenter and Dowsland (1985) developed and discussed three stability criteria for the pallet loading problem as:

• Supportive criterion: the level of criterion is defined by the percentage area required to be in contact with two or more supporting boxes. In this case each box must have its base in contact with at least two boxes in the layer below, for which contact of less than 5% of the box's base area will be ignored.

• Base contact criterion: Each box must have at least 75% of its base area in contact with the layer below. This criterion was applied with the percentage of the contact area set values between 95% to 75%. Stacks which contain boxes with less than 75% of their base area in contact with layer below showed susceptibility to crushing or toppling

• Non-guillotine criterion: There must be no straight or jogged (+/-0 mm) guillotine cuts traversing more than 100% of the stack's maximum length or width. When applying this criterion, two variables must be specified; the precision with which a guillotine cut is defined (I mm), and the length of a cut as a portion of the stack's maximum length or width (Z%) With I=0 mm any cut must be exactly vertical, and considering the case where such a cut must be extended over the whole of the stack Z=100%. Then 91% of the pallet combinations proved to be stable. Therefore, there must be no exact guillotine cut extending above the whole stack dimensions, Z=100%, & I=0 mm.

Note that in some practical situations, where the weight of the transported product is not distributed over the box base, criterion 1 may not be suitable to use. However it might be necessary to apply it to the corner boxes in a stack, since those boxes are most critical.

The specifications of the criterion mentioned, by Carpenter and Dowsland, are:
The first criterion ensures that columns of boxes that have little or no "interlock" with the remainder of the stack are not created, since such a column is potentially unstable when the pallet is transported. The second criterion eliminates the situation where a box is not supported over most of its base. Since this would result in its failing to meet the stationary stability requirement. The third criterion considers the problems associated with guillotine section cutting the pallet stack in a vertical direction. Similar to the columns of boxes, such sections may be at risk during transportation.

CHAPTER 5 PROPOSED MODELS

The research deals with the three-dimensional stacking palletization problem of multi-size boxes. Three models are presented in the following sections. Model 1 is an Integer Linear Programming model (ILP), which is discussed at first. This model modifies the ILP model developed by Tsai et al (1988) and expands that from 2D to 3D pallet volume optimization. The boxes, in the physical layout for this problem, can be loaded either with the layer or stacking procedures Model 2 is developed to complement the proposed Model 1, for which a stacking procedure can be obtained, this model is an ILP as well. Unlike Model 1, that does not indicate the location of boxes on the pallet, Model 2 provides a pre-determined locations for boxes on the pallet. Model 1 and Model 2 deal with boxes of multi-size and both are suitable for the situations in which all the information about boxes dimensions, the number and availability of types of boxes are known. However, they can not be applied to the situations where the arrival of boxes of multiple size is in a random pattern sequence. For such a case, no method is known to obtain the optimal solution for 3D stacking palletization problem Therefore, the results obtained are infeasible by computation. For these reasons, and to experience the effect of the stochastic process on the palletization, Model 3 is developed to accommodate the situations where arriva' of boxes is in a random pattern sequence. Model 3 employs the heuristics approaches to maximize the pallet volume utilization, stability of the column stacks, and to minimize the WIP area and the palletization time.

5.1 Model 1: A 3D ILP Pallet Volumetric Optimization

In this section ILP procedure is developed to deal with the three-dimensional pallet loading problem. The procedure optimizes the volume utilization of the pallet with small rectangular boxes of multiple size. The same idea and procedures may be applied for 3D cutting-stock, packing, and allocation problems. The problem can be classified as follows:

• the volumetric pallet dimension is known and fixed, which is L*W*H,

• boxes are all of multiple size and of both rectangular and square shapes,

• the number and the availability of the boxes are not restricted,

• the demand restrictions on the box quantity is set,

This problem can be characterized as follows:

• the three-dimensional mathematical ILP problem is generated,

• the total volume of boxes must not exceed the pallet volume,

• the upper bound for the pallet base dimension is set such that no over hanging is allowed, hence the total area of the boxes must be less than or equal to the pallet area.

The specifications of the mathematical model to be discussed are as follows:

• there are N type of original boxes of dimension l*w*h, for i=1,..,N

• boxes are allowed to rotate about Z axis, hence the boxes can have two orientations on the pallet, with their height being parallel to the pallet height. This means that each box is considered twice on the pallet except for the case where the length and width of the box are equal,

- the volume of the pallet is pre-determined,
- boxes are all of different dimensions,
- the number and the availability of boxes of each type are finite and known,
- mathematical ILP procedure is considered.

The solution to the problem is achieved by the mathematical model as:

An ILP model is developed to determine the volumetric optimal layout for the pallet The objective function is:

$$Maximize \left\{ \sum_{i=1}^{N} l_i*w_i*h_i*X_i + \sum_{i=N+1}^{2N} \hat{a}_i*l_i*w_i*h_i*X_i \right\} \qquad (5.1.1)$$

whre;

$$\begin{cases} \hat{a}_i = 0, \text{ if } l_i=w_i \text{ of the box type "i" , for i=N+1,...,2N} \\ \hat{a}_i = 1, \text{ otherwise.} \end{cases}$$

$l_i*w_i*h_i$ is the volume of the box type "i",

X_i denotes the number of boxes of type "i" for i=1,2,..,2N,

The pallet volume utilization is obtained from:

$$U = \frac{\sum_{i=1}^{2N} \hat{a}_i*l_i*w_i*h_i*X_i}{L*W*H} \qquad (5.1.2)$$

A good optimization could be obtained not only by defining the proper objective functions but also the practical constraints that the functions are subjected to. The constraints are the most important elements in any optimization, poor constraints develop a bad Linear Programming and lead to the infeasible solutions, where proper and good constraints results in feasible solutions. The following practical constraints are reflected in the proposed model:

(1) Pallet Volume Constraints

The total volume of the pallet must be greater than or equal to the total volume of the boxes loaded on the pallet. Also all the selected boxes must fit within the pallet dimension. This has been considered in the strip constraints.

$$L*W*H - \left\{ \sum_{i=1}^{N} l_i*w_i*h_i*X_i + \sum_{i=N+1}^{2N} \hat{a}_i*l_i*w_i*h_i*X_i \right\} \geq 0 \qquad (5.1.3)$$

(2) Box Quantity Constraints

A new group of boxes are considered with respect to their base dimension. Two orientations are considered for each box type on the pallet if the length and width of the box are not equal. The demand restrictions on the availability of each box type with respect to their base area is as follows:

$$X_i + \hat{a}_i*X_{i+N} \leq D_i ; \qquad \forall i \qquad (5.1.4)$$

Where, D_i is the number of boxes with equal base area of type "i" and type "i+N" available.

(3) Stability Constraints

The stability constraints are not considered in the formulation of this algorithm, since the objective is the volumetric optimization of the pallet. However, the stability is considered in the practical case after the optimal values are obtained by the mathematical model, such that the boxes with larger base dimension are loaded on top of the smaller ones

(4) Boxes of Unit width and length

Boxes of unit length are considered in the generation of the strips types. However, this box is not used in the strip constraints. The length "1" is considered in the box length set "s" and the generation of possible types of strips, since the generation of some strip types combinations can not be made if the number of "1" elements in a strip exceeds the limited unit length. Since the pallet of length L*W is chopped into W strips of dimension L*1, then the maximum allowable unit length in each strip combination would be the smallest dimension of the pallet. Hence, the maximum number of unit length to be used in each strip constraint must not exceed the width of the pallet. The upper bound on the boxes of unit length is:

$$W - X_{2N+1} \geq 0 \qquad (5.1.5)$$

W is the pallet width,

X_{2N+1} is the number of boxes of unit area.

(5) Strip Constraints

At first general form of strip types is considered. In order to formulate the pallet loading problem, as a linear programming problem, all the possible strip types combinations containing the length of the boxes must be determined. Since each box can have two different orientations, assign the box length set which is a combination of the boxes lengths and widths to $S=\{l_1,w_1,l_2,w_2,\ldots\ldots,l_N,w_N,l_{N+1},w_{N+1}\}$.

where $l_{N+1}=w_{N+1}=1$ represent the length and width of the box of unit area respectively.

The combinations of lengths to form all the possible types of strips must be considered. However, since two elements in the length set "S" could have the same value, a new set must be defined such that its elements are unique. The new set "s" is the largest subset of "S" representing the box length set with unique elements, whose elements are used to generate the possible strip types. This length set can be shown as:

$$s=\{l_1^*, l_2^*, l_3^*,\ldots, l_k^*, l_{k+1}^*\} \tag{5.1.6}$$

where;

$l_1^* \neq l_2^* \neq l_3^* \neq \ldots\ldots \neq l_k^* \neq l_{k+1}^*, \quad l_{k+1}^* = 1, \quad k \leq 2N.$

"s" is the largest subset of length set "S", when all the subset elements are unique, and are used to generate the possible strip types.

l_e^* is the length of a box, which is selected from the length set "S" and $l_e^* \neq l_f^*$ if $e \neq f$.

Hence, all the possible types of strips can be expressed in the following equation:

$$a_1 l_1^* + a_2 l_2^* + a_3 l_3^* + ...+ a_e l_e^* + ...+ a_k l_k^* + a_{k+1} l_{k+1}^* = L, \qquad (5.1.7a)$$

where, e=1,2,3,....,k+1

Therefore, in general, for *n* possible types of strips, all the possible box length combinations of which their sum is equal to the pallet length can be expressed as,

$$j\ ) = \{\ a_{ej}\ |\ where \sum_{e=1}^{k+1} a_{ej} * l_e^* = L;\ e=1,2,..,k+1,\ \forall j\ \} \qquad (5.1.7)$$

where; "j" is the type of strip on the pallet, for *n* possible types of strips, j=1,2,...,*n*

a_{ej} is the number of pieces (lengths) of length l_e^* in type j strip,

l_e^* is the length of box which is selected from the length set "S", 1< e < k+1,

Hence, the following strip constraints are determined:

For j=1,2,...,n types of strip from equation (5.1.7), let y_1, denote the number of the strips of type 1, y_2, denote the number of strips of type 2, etc., and y_j, denote the number of strips of type "j". Then the sum of number of strips of each type "j" times the pallet height H must be less than or equal to the pallet width W times the pallet height H, but, the height H cancels from the both sides. Therefore, we have the following constraints;

$$\sum_{j=1}^{n} y_j \leq W\ ; \qquad j=1,2,...,n \qquad (5.1.8)$$

The other strip constraints are defined as follows, with respect to the length set "s":

For length l_e^*, e=1,2,...,k+1.

$$\sum_{j=1}^{n} a_{ej}*H*y_j \geq \sum_{i=1}^{N} w_i*h_i*X_i + \sum_{i=N+1}^{2N} \alpha_{e,i}*w_i*h_i*X_i \; ; \quad \forall e \qquad (5.1.9)$$

$$\begin{cases} \alpha_{ei} = 0, & \text{if } l_i = w_i, \text{ for } i=N+1,\dots,2N \\ \alpha_{ei} = 1, & \text{otherwise.} \end{cases}$$

If the length and width of a box are equal, then for that type of box no rotation about the Z axis is considered. Hence, $X_{i+N} = 0$, if $l_i=w_i$; i=1,2,..,N original box type.

Note that, no over hanging is allowed when loading boxes on the pallet, such that the area of the total boxes loaded on the pallet and on each layer on the column stacks must be less than or equal to the pallet area. And since we deal not only with the area of the boxes but also with their length and width, the total length and width of the boxes laying along the pallet length must be less then or equal to that length L. Also the total length and width of the boxes laying along the pallet width must be less than that width W. The length over hanging problem is solved through strip types equation (5.1.7), for which the sum of elements (lengths) in each strip is equal to the pallet length L. The width over hanging problem is solved through the constraint in equation (5.1.8), for which sum of all possible types of strips y_j is less than or equals the pallet width W. Furthermore, the total height of the boxes stacked on top of each other in each column stack on the pallet is restricted to the maximum palletization height H. This problem is solved by using the pallet volume constraint equation (5.1.3).

5.1.1 The ILP formulation for Model 1 Problem

The general form of the formulation for the Model 1 problem can be shown as follows, where the objective function in terms of pallet volume maximization is:

$$Maximize \left\{ \sum_{i}^{N} l_i * w_i * h_i * X_i \; + \sum_{i=N+1}^{2N} \hat{a}_i * l_i * w_i * h_i * X_i \right\}$$

Subject TO:

$$L*W*H - \left\{ \sum_{i=1}^{N} l_i * w_i * h_i * X_i \; + \sum_{i=N+1}^{2N} \hat{a}_i * l_i * w_i * h_i * X_i \right\} \geq 0$$

$$W - X_{2N+1} \geq 0$$

$$X_i + \hat{a}_i * X_{i+N} \leq D_i \; ; \qquad \forall i$$

$$\sum_{j=1}^{n} y_j \leq W \; ; \qquad j=1,2,\ldots,n$$

$$\sum_{j=1}^{n} a_{ej} * H * y_j \geq \sum_{i=1}^{N} w_i * h_i * X_i \; + \sum_{i=N+1}^{2N} \alpha_{ei} * w_i * h_i * X_i \; ; \quad \forall e$$

$$\mathbf{X_i} \geq 0 , \quad \forall \, i$$
$$\mathbf{X_i} \text{ integer}$$

The optimal values obtained from the formulation above provide solution to the volumetric 3D pallet loading problem. After the optimal values are obtained, the decision about how and where the boxes will be placed on the pallet can be made, considering the stability criterion for the column stacks.

5.1.2 Algorithm for Model 1

The optimal result to the 3D pallet loading problem, for which a direct ILP programming to maximize the volume utilization of the pallet is used, can be determined from the following steps:

1- input the number and the type of boxes, indicating their volumetric dimension, l_i,w_i,h_i,

2- input the pallet dimension, length, width and the stacking height,

3- initialize,

4- define the box length set "s", with its unique element.

5- generate the strip types with respect to the unique length and width of the boxes in the box length set "s",

6- set the required constraints, equations (5.1.3) to (5.1.9),

7- develop the ILP formulation shown in section 5.1.1,

8- solve the ILP problem,

9- obtain the optimum values for the problem

10- if the values are satisfactory go to next step, or else go back,

11- calculate the pallet volume utilization,

12- present the optimal values on the computer,

13- simulation, considering the stability of column stacks,

- load the boxes on the pallet, step by step,
- draw and present the final results on the computer,

14- End.

5.2 Model 2: Stacking Optimization for the 3D Palletization.

The Model 2 is developed to obtain solutions to the 3D stacking palletization, for which the location of the boxes to be loaded onto the pallet is pre-determined, since the Model 1 does not indicate the location of the boxes on the pallet.

Many have shown that the mathematical model for the one and two-dimensional cutting-stock problems result in optimum values. Such algorithms can be found in the papers presented by: Gilmore and Gomory (1961-63-65), Haessler (1975), Chambers (1976), Dyckhoff (1981), Tsai et al. (1988), Farely (1988-90),... Hence, the combination of the mathematical model of the one and two-dimensional cutting-stock problems can lead to the development of a mathematical model for the three-dimensional pallet loading problem. Model 2 employs an ILP procedure to 2D surface partitioning, resulting to some optimal subareas (small partitioned area) on the pallet, then multi-layer stacking is considered for each optimal subarea until the stacking height is achieved in all the columns.

The problem classifications are:

- dimension of the pallet L*W is fixed and known,
- the stacking height on the pallet is pre-determined,
- boxes dimension are known as well as number and the availability of boxes of each type,

• the demand restrictions on the boxes base dimension, and their respective height is predetermined.

Furthermore, this type of stacking palletization is characterized as follows:

• the two-dimensional mathematical ILP problem is generated,

• the optimal layout for the two-dimensional pallet loading is determined, providing the base dimension of the boxes available,

• there can be number of similar rectangular bases on the pallet,

• the upper bound for the pallet area utilization is set, such that no over hanging is allowed,

• the pallet stacking height is pre-determined,

• the height of the stack in each optimum subarea is restricted to the pallet allowable stacking height,

• the multi-process of multi-layer stacking ILP problem is generated,

• the number of boxes in each column stack, even in those with the same base dimension, may differ from one another, since the height of the boxes may be different,

• the stacking height in each column stack may differ,

The mathematical problem to be discussed has the following specifications:

• there are G types of original boxes, of cubic dimension l*w*h, for ih = 1,2,...,G,

• there are N types of original boxes base area, i=1,2,...,N, type 1,.., type i,.., type N,

• there are P types of original heights, corresponding to the base type "i", for h=1,2,...,P,

• the boxes are of different dimensions,

• since the number and availability of boxes type "ih" is finite and known, we have

  • the number and availability of boxes base of type "i" (D_i) is finite and known,

  • the number of heights with respect to boxes base is known, D_{ih},

• the dimension of the pallet and the stacking height are fixed,

• boxes are allowed to rotate about the Z axis, so that each box can have two different orientations on the pallet. Hence, the number of boxes base-area is doubled to 2N,

• multi-stacking palletization is considered,

• the maximum volume of the palletization is pre-determined. However, the height of column stacks not exceeding the pallet stacking height may differ from each other This may result in lower percentage of volume utilization of the pallet,

• mathematical ILP procedure is considered.

The solution to the problem is achieved as follows:

An ILP model is developed with two interactive objective functions, for which the second one depends on the result of the first one The initial objective function considers the two dimensional pallet loading problem, such that the area of the pallet is maximized by the rectangular pieces corresponding to the boxes base dimension. In this case, applying the guillotine- or the nested-cut pattern, the optimum layout for the pallet area is determined mathematically. The initial objective function is;

$$\textit{Maximize} \sum_{i}^{2N} \hat{a}_i * l_i * w_i * X_i \qquad (5.2.1)$$

where, $\hat{a}_i=0$, if $l_i=w_i$, for $i=N+1,..,2N$, $\hat{a}_i=1$, otherwise.

X_i denotes the number of rectangular pieces (corresponding to the boxes base area) l_i*w_i of type "i", for i=1,2,. ,N,

N denotes the number of original box type,

The pallet area utilization is obtained from:

$$UA = \frac{\sum_{i}^{2N} \hat{a}_i*l_i*w_i*X_i}{L*W} \tag{5.2.2}$$

After the optimum layout, optimum subarea, on the pallet is determined from the initial objective function, assign each optimum base area found to a new index:

Z_j denotes the number of optimum subarea (boxes base) "j" found on the pallet, for j=1,2, .,T

Z_{jh} denotes the number of boxes of optimum base type "j" and height "h".

Then, the specific column corresponding to each subarea is maximized. This procedure is considered as multi-process stacking problem with multi-layer in each column stack. The objective function to optimize the stacking height for each subarea is:

$$Maximize \sum_{j=1}^{\sum_{k=1}^{T} n_k} \sum_{h=1}^{P} h_h'*Z_{jh} \tag{5.2.3}$$

where, T is determined from initial ILP problem, which denotes the number of different optimum types of subarea found on the pallet,

n_k denotes the number of times each optimum base (subarea) type "T" is repeated on the pallet, for k=1,2,...,T,

h'_h denotes the height of boxes of height type "h", in ascending order for index h=1,2, ..,P, such that i.e. $h'_1=1$, $h'_2=2$, $h'_3=3$.

The most essential part of the ILP formulation which conducts the solutions to the objective function are the constraint. The following practical constraints are reflected in the proposed model:

(1) Pallet Constraints

When loading the boxes onto the pallet no over hanging is allowed, therefore, the total surface area of the pallet must be equal to or greater than the total area of the boxes on the pallet surface, regardless of their heights.

$$L*W - \sum_{i=1}^{2N} \hat{a}_i*l_i*w_i*X_i \geq 0 \qquad (5.2.4)$$

(2) Stacking Height Constraints

The total area on the pallet is partitioned into several optimal strips/subareas, and each column stack corresponding to the optimal subareas contains few boxes of different or similar height. Therefore, the total stacking height of boxes in each strip must be less than or equal to the maximum palletization height (pallet height) H.

$$H - \sum_{h=1}^{P} h_h' * Z_{jh} \geq 0 \ ; \quad \forall j \tag{5.2.5}$$

For example, if there are three different heights, h_1, h_2, h_3, a number of layer combinations can be generated. If the number of layers of the same height on the pallet are denoted by a, b, and c respectively, the following constraint must be satisfied:

$$a*h_1' + b*h_2' + c*h_3' \leq H \tag{5.2.6}$$

The total number of layer combinations (a,b,c) is obtained from the number of solutions of this inequality. In general, if there are N different heights $h_1, h_2, .., h_i, .., h_N$ the number of combinations of layers on each strip can be generated. Let X_{ij} denote the number of layers of the same height on each subarea, therefore, the following constraint must be satisfied:

$$H_j - \sum_{i=1}^{N} X_{ij} * h_i \geq 0 \ ; \quad \forall j \tag{5.2.7}$$

The number of solutions to this inequality give the total number of layer combinations (X_j, X_{2j},..., X_{nj}). Each layer combination is considered as an alternative for the palletization problem. So, the solution to the original problem is obtained from one of the alternatives.

(3) Box Quantity Constraints

The number of boxes of each type available is limited and each box can have two different orientations on the pallet, considering the base dimension of each type of box

$$X_i + \hat{a}_i * X_{i+N} \leq D_i \; ; \qquad \forall i \qquad (5.2.8)$$

Where; D_i is the number of boxes base area of type "i" and type "i+N" available.

The number of different heights with respect to the base dimension of boxes of each type is finite. Hence, the constraints on the availability of the boxes' height is;

$$\sum_{j=e_k}^{f_k} Z_{jh} \leq D_{jh} \; ; \quad \forall k \text{ optimum type} \; , \; \forall h \qquad (5.2.9)$$

$$\text{where; } e_k = 1 + f_{k-1}, \; f_k = n_k + f_{k-1}, f_0 = 0$$

D_{jh} is the number of boxes of optimum base dimension of type "j" and height "h".

(4) Upper Bound Constraints (for number of boxes of unit area)

Consider a two-dimensional pallet loading problem, with boxes of different dimensions, where all the dimensions are considered to be integer, divide the pallet of dimension L*W into W strips of dimension L*1. Assuming that the boxes can have two different orientations on the pallet, it does not guarantee that the length of each strip L*1 will be filled 100%, by combination of boxes length l_i and width w_i. Therefore, some boxes of unit area, of length and width one can be considered to provide at least one optimal solution to the total coverage of the pallet. For the proposed model, the upper bound (maximum number) of boxes of unit area $\{\max(a_{k+1})\}$ is set less than or equal to the smallest dimension of the pallet, which is the pallet width W.

$$\max(a_{k+1}) \leq W \tag{5.2.10}$$

Lat X_{2N+1} be the box type of unit area, $l_{2N+1}=w_{2N+1}=1$, Then

$$\begin{aligned} \max(a_{k+1}) - X_{2N+1} &\geq 0 \\ W - X_{2N+1} &\geq 0 \end{aligned} \tag{5.2.11}$$

(5) Strip Constraints

The strip types combinations for this model are similar to those generated in the Model 1 problem, however their application is slightly different. Boxes are of different dimensions, the box length set "s" is obtained similar to that of equation (5.1.6), with its unique elements. To generate the strip types combination, for n possible types of strip, all the possible box length combinations of which their sum is equal to the pallet length L are obtained exactly as those in equation (5.1.7).

The following strip constraints are determined:

For j=1,2,...,n types of strip from equation (5.1.7), let y_1 denote the number of the strips of type 1, y_2 denote the number of strips of type 2, and y_j denote the number of strips of type "j", etc. Then the sum of the number of strips of each type "j" must be less than or equal to the pallet width W.

$$\sum_{j=1}^{n} y_j \leq W ; \qquad j=1,2,...,n \tag{5.2.12}$$

The other strip constraints are defined as follows, with respect to the length set "s":

For the length l_e^*, e=1,2,...,k+1

$$\sum_{j=1}^{n} a_{ej}*y_j \geq \sum_{i=1}^{N} w_i*X_i + \sum_{i=N+1}^{2N} \alpha_{e,i}*w_i*X_i + \alpha'_{e,i}*X_{2N+1} \;; \quad \forall\, e \qquad (5.2.13)$$

where;

$\alpha'_{ei} = 0$, if $l_e \neq l_{k+1}$

$\alpha'_{ei} = 1$, otherwise. Thi· represents the unit 1*1,

$l_{k+1} = 1$,

X_{2N+1} = box type of unit area, $l_{2n+1} = 1, w_{2n+1} = 1$,

a_{ej} is the number of lengths of length type l_e^* in type j strip,

$\alpha_{ei} = 0$, if $l_i = w_i$, for i=N+1,...,2N

$\alpha_{ei} = 1$, otherwise.

If the length and width of a box are equal, then for that type of box no rotation about the Z axis is considered. Hence,

$X_{N+i} = 0$, if $l_i = w_i$; i=1,2,..,N original box type.

An example to illustrate the use of strip constraints is provided on the following pages.

(5a) Example (strip constraints):

Constraints of this type have been developed from Brown's linear equation approach. This algorithm guarantees that the selected boxes will fit the pallet area with a specific orientation. The following example is considered to describe the linear equation approach.

Consider a collection of eight small boxes with respective length and width of their base dimensions of (3,2), (3,1), (3,1), (2,3), (2,2), (1,3), (1,2) and (1,1). These small boxes are to fit into a large pallet of dimension (7,4), representing its length and width respectively. The height of the boxes is discarded at this moment. The total sum of areas for these eight boxes is exactly equal to the area of the pallet which is 28. Therefore, these eight boxes have to completely fill the pallet area. The lengths and widths of boxes are considered to be parallel to the length and width of the pallet respectively. The orientation of the boxes is considered to be fixed. Note that if the area is the only restriction to be considered the problem is simplified, and the assigned boxes can fill the pallet area. However, the boxes are considered not only by area but also by their lengths and widths.

The box length set is the combination of the boxes length and width, these lengths and widths have to be unique in order to present the length set, which is assigned to s={3,2,1}. The linear equation techniques can be used to check whether this combination of the box lengths presented in the set "s" can fit into the pallet area of dimension 7x4. The procedure is as follows:

First divide the pallet into four strips of dimension 7x1, and then consider all the multi-set combinations of the lengths that can fit and fill the strips of length 7 on the pallet. The feasible combinations to fill these strips are:

a) {3,3,1}

b) {3,2,2}

c) {3,2,1,1}

d) {2,2,1,1,1)

e) {2,2,2,1}

f) {3,1,1,1,1}

g} {2,1,1,1,1,1}

However, since there are only two boxes of the length 2, and three boxes of length 1, the combination sets of {2,2,2,1}, {3,1,1,1,1} and {2,1,1,1,1,1} can not be possible strip types. Hence, the only possible strip combinations are strip types (a), (b), (c), and (d). Let "a" denote the number of the strips combination of type (a), "b" the number of strips combination of type (b), etc. Where the number of boxes of type 1, type 2, type 3, etc. on the pallet can be represented by X1, X2, X3, etc. respectively. There are totally four strip types, thus we can obtain the linear equation;

$$a + b + c + d \leq 4 \qquad \text{(e1)}$$

There are two types of boxes of length 3, namely (X_1=3*2, X_2=3*1,X_2=3*1) for which one has width 2, and two have width 1. Also, there must be four strips of length 3, two in each type (a) strip, one in each type (b) strip and one in each type (c) strip. This provides the linear equation;

$$2a + b + c \geq 2X_1 + 2X_2 \qquad \text{(e2)}$$

For the boxes of length 2, the same idea is applied. There are two types of boxes of length 2, one of width 3, and one of width 2. There must be five strips of length 2, two in each type (b) strip, one in each type (c) strip, and two in each type (d) strip. This yields the linear equation;

$$2b + c + 2d \geq 3X_3 + 2X_4 \qquad \text{(e3)}$$

Similarly, there are three types of boxes of length 1, one of width 3, one of width 2, and one of width 1, for which there is one in each type (a), two in each type (c) and three in each type (d). The following linear equation is obtained;

$$a + 2c + 3d \geq 3X_5 + X_6 + 2X_7 \qquad \text{(e4)}$$

The linear equations above become constraints when X1, X2, X3, etc. are decision variables in a model. The number of strip constraints increases with the increase of the number of boxes Solving the above linear equations (el) through (e4), the integer solution obtained is a=1, b=1, c=1 and d=1. Hence, the combinations of the areas of the boxes will fill the pallet area.

5.2.1 The ILP Formulation for Model 2 Problem

The ILP Formulation for Model 2 problem for which the optimum layout on the pallet can be determined is shown as:

$$Maximize \sum_{i}^{2N} \hat{a}_i * l_i * w_i * X_i$$

Subject TO:

$$L*W - \sum_{i=1}^{2N} \hat{a}_i * l_i * w_i * X_i \geq 0$$

$$X_i + \hat{a}_i * X_{i+N} \leq D_i \quad ; \quad \forall i$$

$$W - X_{2N+1} \geq 0$$

$$\sum_{j=1}^{n} y_j \leq W \quad ; \quad j=1,2,\ldots,n$$

$$\sum_{j=1}^{n} a_{ej} * y_j \geq \sum_{i=1}^{N} w_i * X_i + \sum_{i=N+1}^{2N} \alpha_{e,i} * w_i * X_i + \alpha'_{e,i} * X_{2N+1} \quad ; \quad \forall e$$

$$X_i \geq 0 \; ; \quad \forall i$$
$$X_i \text{ integer}$$

The optimum value obtained from the above formulation must be used to develop the next ILP problem, in order to determine the pallet optimum stacking height for each subarea/strip. Therefore, the second objective function to maximize the stacking height on each optimal subarea on the pallet can be formulated as follows:

$$\textit{Maximize} \quad \sum_{j=1}^{\sum_{k=1}^{T} n_k} \sum_{h=1}^{P} h_h^{/} * Z_{jh}$$

Subject TO:

$$H - \sum_{h=1}^{P} h_h^{/} * Z_{jh} \geq 0 \quad \forall j$$

$$\sum_{j=e_k}^{f_k} Z_{jh} \leq D_{jh}; \qquad \forall k \ \textit{optimum type} \ , \forall h$$

$$\textit{where;} \quad e_k = 1 + f_{k-1}, \quad f_k = n_k + f_{k-1}, \quad f_0 = 0$$

$$Z_{jh} \geq 0, \ Z_{jh} \ \textbf{Integer}$$

For instance; Let "T" represent the number of different optimum bases found, and n_k denote the number times the optimum type T is repeated, for k=1,2,...,T. Where, h=1,2,...,P is the different heights, then n_k*p represents the number of boxes of type "k" which have the same base dimension, and their heights are h=1,..,P. Assume that two types of optimal base area are obtained from the first ILP problem. If the optimum type one is repeated two times, and type two is repeated two times as well, then n_1=2, n_2=2. Also consider that there are three different heights for each type, then;

h=1,..,3 so that P=3. Therefore, n_1*P = 2*3 = 6, which means the first six types of box Z_{kh} (k=1,..,T ; h=1,..,P) have same base dimension. The first six types of box are Z_{11},Z_{12},Z_{13},Z_{21},Z_{22},Z_{23}. Where; $Z_{11}=Z_{21}$, $Z_{12}=Z_{22}$, $Z_{13}=Z_{23}$.

Also, n_2*p = 2*3 = 6, which means the second six types of box Z_{kh} (k=n1+1,..,n1+n2 ;

h=1,..,P) have same base dimension. The second six types of box are $Z_{31},Z_{32},Z_{33},Z_{41},Z_{42},Z_{43}$. Where; $Z_{31}=Z_{41}$, $Z_{32}=Z_{42}$, $X_{33}=Z_{43}$

Note that the solutions to a problem varies as the demand restrictions on the availability of the boxes of each type varies. These demand restrictions can be set both when determining the optimum layout for the pallet area and when determining the optimum layout of stacking height. Several examples are conducted to verify the changes on the results obtained with respect to different demand restrictions set in the first layout of the ILP problem for dimensions L*W, and also in the layout of the stacking heights corresponding to those optimum results.

5.2.2 Selection of Column Stacks

The selection of column stacks is critical in the pallet loading problem, and it may be concerned with respect to two major concepts. Stability and utilization. When loading with boxes of different heights, the stability of the column stacks become one of the major factors in the palletization problem. Since the larger the box the more stable it is, the boxes with bigger volume, therefore, would be loaded on the smaller boxes in each column to provide better stability.

In order to maximize the pallet utilization, with respect to the demand restrictions set for the problem, the boxes with larger base area have the priority to be loaded on the pallet.

This provides a higher percentage in the pallet area utilization, and also provides a better chance for the smaller boxes to be loaded on the top of the bigger ones. However, where the stability of column stacks is under question the larger box must be loaded on the top. Since the objective is the utilization, after the optimum layout for the pallet area is determined, the larger optimum subareas have the priority to stack up, so that the volume utilization of the pallet is maximized. By convention boxes are loaded on the bottom left hand corner on the pallet. If the optimum type of boxes is found to be repeated a number of times on the pallet, the procedure of the column selection remains the same. Moreover, the optimum stacking heights for each optimum subarea is determined, then, for each optimum type subarea (starting with the larger ones), the boxes with greater height will be loaded first. This provides a greater volume utilization of the pallet. The algorithm for this model is given in the following pages.

5.2.3 Algorithm for Model 2

The following steps are to determine the solutions to the 3D stacking optimization problem, by ILP procedure, for Model 2 problem:

1- input the number and the type of boxes,

2- input the pallet(s) dimension and the stacking height,

3- initialize,

4- extract the base dimension of each type of box and their corresponding heights,

5- generate the strip combinations with respect to the unique lengths and widths of the boxes,

6- set the required constraints,

7- set the demand restriction constraints on the availability and the number of boxes of each type,

8- develop the ILP formulation

a) for the two-dimensional pallet loading to maximize number of boxes on the pallet,

- set the upper bound for the pallet dimension, and strip combinations,
- set the upper bound for the unit pieces,

b) for the multi-process multi-layer stacking problem to maximize the stacking height in each column, with respect to the optimum subareas found,

9- solve the ILP problem,

10- if solved 8-a)

- find the optimum subareas on the pallet,

- if the solution is not acceptable, verify the strip combinations and the ILP formulations, go to step 5, or else go to step 8-b.

- if solved 8-b)

  - find optimum stacking height in each column,

  - if the solution is not acceptable and there is residual height on each column stack, go to 11, else go to 15.

11- determine the remaining height and the optimum base area,

12- consider each unfilled base area as a small pallet,

13- set the stacking height to the corresponding residual height on each subarea,

14- set the number and the availability of each type of box to the remaining appropriate boxes,

go to step 8-a).

15- calculate the pallet area utilization,

16- calculate the pallet volume utilization,

17- present the values on the computer,

18- simulation

  - draw the optimum layout on the pallet area,

  - draw the stacking height on each column stack,

  - present the final results,

19- End

The flow chart for this model is shown in figure 5.2.1.

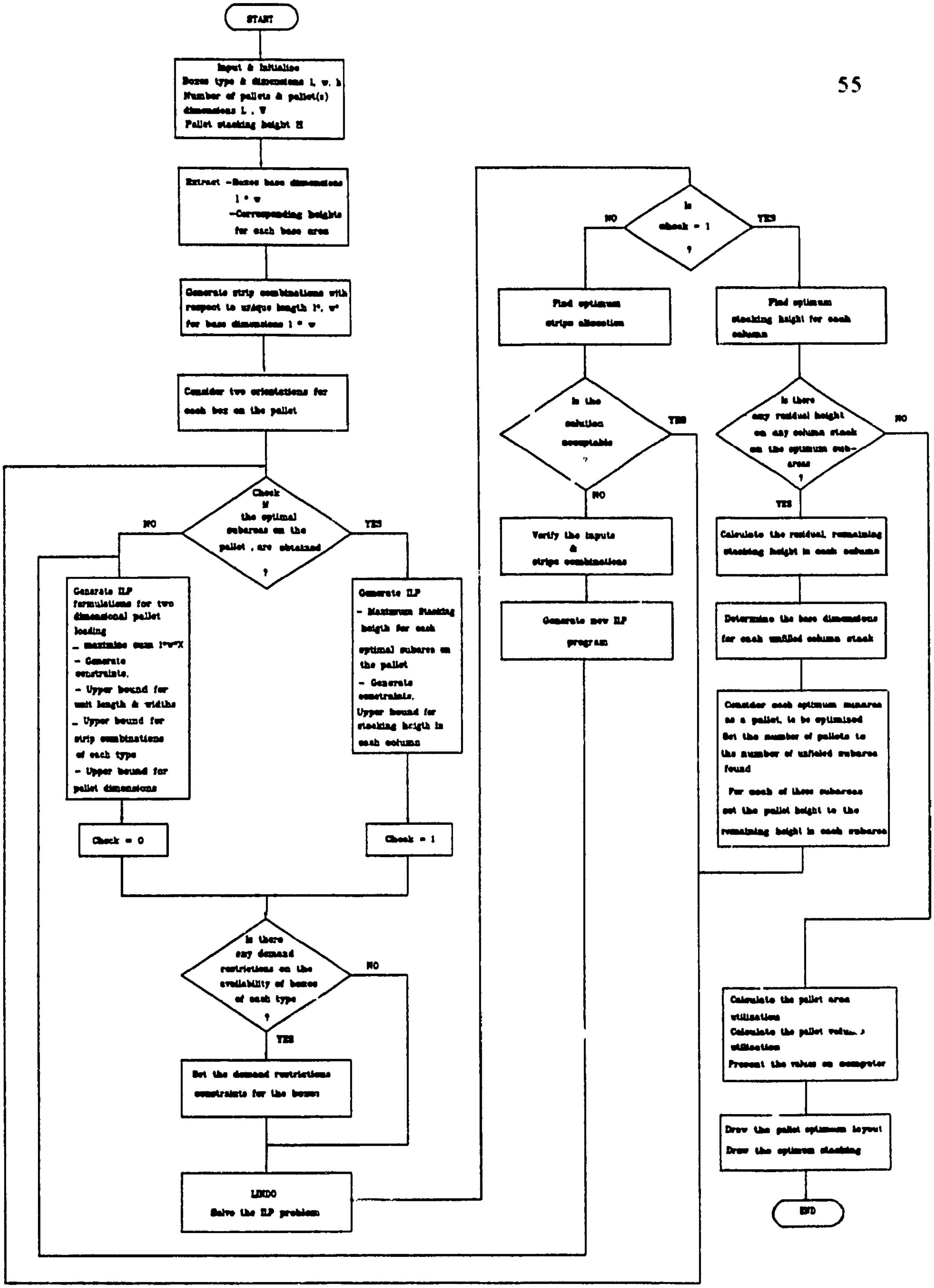


Figure 5.2.1 Flow Chart for 3D Stacking for Model 2

5.3 Model 3: A Random Sequence Heuristic for 3D Stacking Palletization

In automated manufacturing systems the palletization takes place at the end of production lines. The arrival of boxes is in a random sequence pattern, so, the availability of each type of box is not a constant. The solutions to such a palletization problem, with stochastic process, can be infeasible by computation therefore, heuristics are required. The heuristics approaches with some practical assumptions are appropriate to deal with complex optimization problems, for which algorithms yielding optimal solutions are not known or infeasible by computation. So far, no mathematical model, such as those proposed earlier in Model 1 and Model 2, has been developed to solve such a complex problem. There is no literature on the heuristic for 3D stacking palletization problems. The proposed Model 3 applies a heuristic to deal with problems with the following specifications:

- one pallet, or more if necessary, with fixed dimensions,
- number of boxes types is finite, however their arrival is in a random sequence,
- box are grouped according to their similar height. A group is a collection of boxes with same height,
- there are seven types of boxes, such that three types of base, and for some, three different heights are considered,
- maximum number of different heights is three, hence, the maximum number of groups is three,
- boxes in group 1, 2, and 3 have height of h1, h2, and h3, respectively. Height h1 is the

shortest height and height h3 is the tallest height,

• each group has a holding area,

• the objective is to reach the maximum pallet volume utilization and maintain low WIP,

• stacking palletization technique is applied Stacking is considered from the layout point of view and for each layout stack the layer loading is considered

• there is no restriction on the number of boxes available

• single pallet palletization

In heuristic palletization, the goal of achieving stability does not seem to be realistic since the arrival of certain types of boxes is not certain Therefore, the main goal in most situations is utilization, which is selected as the objective of the proposed model Note that the proposed Model 3 is a base ILP heuristic model. The optimal layout on the pallet with respect to the boxes base area, using LINDO, is determined. Then, each optimal layout on the pallet is stacked up providing layers with different heights in each column stack, such that each column stack may have one or more layers but the height of the stack remains less than or equal to the palletization height. Each layer consists of one or more boxes. The height of each layer in a column remains constant until the layer is filled. Boxes can form blocks and then each block is treated as a box and loaded on a proper subarea on each column stack. The formation of a block is only to ease the loading procedure. The block formation is as follows: A block can contain one or more boxes Smaller boxes are used to form blocks (i e block of dimension 40*24*20 can be formed by four boxes of dimension 20*24*10, or one box of dimension 20*24*20 and two boxes of dimension 20*24*10). As the number of boxes of different dimensions increases the problem of forming block becomes more complex. The block formation is done at the loading time, and may be called partial layer formation. This procedure may be defined as staging procedure for which boxes are selected before loading on the column on the pallet rather than being loaded in terms of first-in first-served.

The solution to the stacking palletization for Model 3, using heuristic approaches, is determined by conducting several steps. These steps are followed in the following section.

5.3.1 Algorithm for Model 3

The following steps are to determine the solutions to the heuristic 3D stacking palletization:

Step 1. Start

Initialize the system. Set the remaining height in each optimal subarea on the pallet equal to the palletization height,

Step 2. Checking the remaining palletization height in subarea "j", H_{jr}, on the pallet. For j=1,2,...,T, being the optimal subareas determined by LINDO,

- if H_{jr} equal to zero, it means all the space in the subarea "j" has been filled,
- then check the next subarea, until j=T, if all zero, terminate the procedure,
- otherwise continue,

Step 3. Checking the new layer on each column stack "j", for j=1,2,..,T

- if a new layer has to be selected, go to step 4,
- if a layer has already been started and waiting to be filled, go to step 5,

Step 4. Selecting the height for new layer

The height of the new layer must be determined before loading any box to that layer. Since the objective of the model is the maximization of each column stack, the priority of the selection of the layer height in each column stack is given to the greatest height possible. Note that there are three heights: h1, h2, and h3, where, h1 is the shortest height and h3 is the tallest height. Also h3 can be formed by combining the height h1 and/or h2 The selection of the height for each layer on each column stack, when boxes are coming to the system in a random pattern sequence, is.

• if there is a box of group 3 coming to the system, the height of the layer in column stack "j", HL_j , is selected to be h3,

• if there is no box from group 3 but there is a box from group 2, then the height of the layer in column stack "j", HL_j , is selected to be h2,

• if there is no box from group 3 and there is no box from group 2 but there is a box from group 1, then the height of the layer in the column stack "j", HL_j , is selected to be h1,

Step 5. Filling a layer in the column stack of the optimum subarea "j" with a box from group "i", which has the selected height h_k for the current layer

The layer in the column stack "j" is filled after the layer height is determined. The height of the layer remains a constant until the completion of the total space in the layer. Blocks are also treated as boxes and then a box/block loaded on the current layer on the column. Furthermore, some rules are followed in the selection of boxes:

• always a larger box is selected prior to the others if it fits the remaining area of the layer,

• select the last box in the pool if it can fit and complete the remaining area of the layer in the column stack "j". Otherwise, leave the smallest box in the WIP area.

• if the new box is too large to fit the remaining area, and there are boxes on the working layer on the column "j" which are smaller then the new box, remove the smaller box to WIP, and restart this step,

Step 6. Checking the layer for completion

• after a layer is completed go back to step 2

Step 7. Checking the height in column stack "j" for completion

• after the remaining height in the column stack "j" is zero, repeat the same procedure for the next subarea,

Step 8. Terminating the procedure

• stop the procedure if all the column stacks are filled and their stacking height are zero

The flow chart of this heuristic is shown in figure 5.3.1.

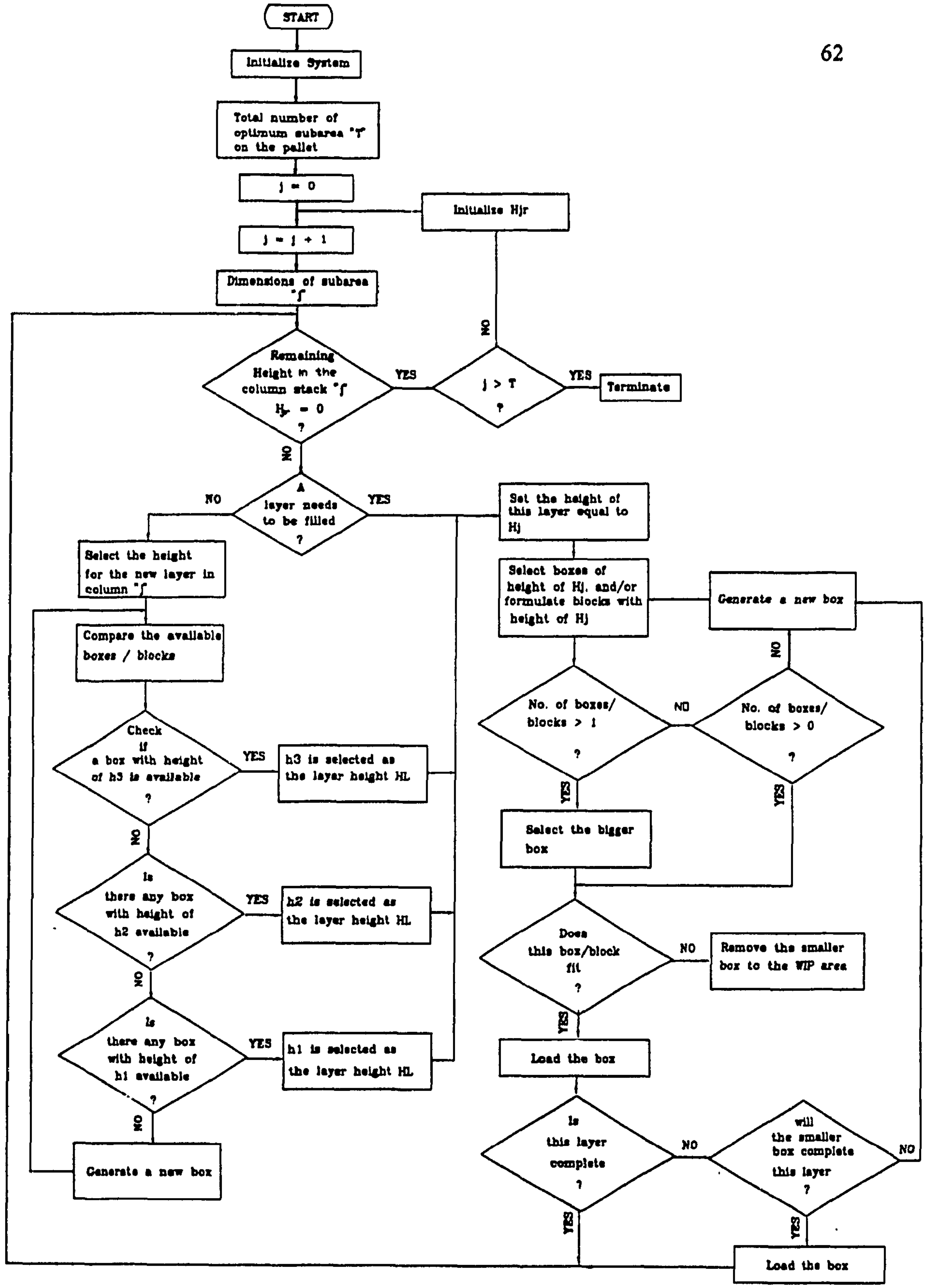


Figure 5.3.1 Flow Chart for 3D Stacking for Model 3

CHAPTER 6 CASE STUDY APPLICATIONS

Several numerical examples have been worked out to demonstrate the validity of the proposed models. Different types of boxes are considered for the proposed Model 1, Model 2, and Model 3, with certain demand restrictions on the availability of boxes of each type. Boxes' dimensions are shown in a few tables presented in each case study.

6.1 Model 1: Case Study

The pallet of dimension L*W*H = 7*4*4, is available. There are six types of original boxes of multi-size, of both rectangular and square shape. These boxes are defined in terms of length, width and height as: B_1=2*2*1, B_2=2*2*2, B_3=2*2*3, B_4=3*2*1, B_5=3*2*2, B_6=3*2*3, B_7=4*2*1, B_8=4*2*2, B_9=4*2*3; table (6.1 1).

Table 6.1.1 Model 1: Input Data (Original Box Type)

Item	Length	Width	Height	Volume
Pallet	7	4	4	112
Box 1	2	2	1	4
Box 2	2	2	2	8
Box 3	2	2	3	12
Box 4	3	2	1	6
Box 5	3	2	2	12
Box 6	3	2	3	18
Box 7	4	2	1	8
Box 8	4	2	2	16
box 9	4	2	3	24

Consider the case for which the boxes are allowed to be loaded on the pallet on any of their side. This leads to the case that boxes rotate about X, Y and Z axes with no restrictions; therefore, in this case each box can be represented in six different forms on the pallet. This number depends on whether the length, width and height of the box are different from each other. The different representations of the boxes of each type are shown in the table (6.1.2).

Table 6.1.2 Model 1: Different Orientations of Boxes

Item	l*w*h	w*l*h	h*w*l	h*l*w	l*h*w	w*h*l
Box 1	2*2*1	-----	1*2*2	-----	2*1*2	-----
Box 2	2*2*2	-----	-----	-----	-----	-----
Box 3	2*2*3	-----	3*2*2	-----	2*3*2	-----
Box 4	3*2*1	2*3*1	1*2*3	1*3*2	3*1*2	2*1*3
Box 5	3*2*2	2*3*2	2*2*3	-----	-----	-----
Box 6	3*2*3	2*3*3	-----	3*3*2	-----	-----
Box 7	4*2*1	2*4*1	1*2*4	1*4*2	4*1*2	2*1*4
Box 8	4*2*2	2*4*2	2*2*4	-----	-----	-----
Box 9	4*2*3	2*4*3	3*2*4	3*4*2	4*3*2	2*3*4

In table (6.1.3) the duplicated type of each box for which boxes are allowed to rotate about the Z , X , and Y axes is shown:

Table 6.1.3 Model 1: Converted Boxes Data

Item	Number	Length	Width	Height	Volume
Pallet	1	7	4	4	112
Box type 1	X1	2	2	1	4
Box type 2	X2	2	2	2	8
Box type 3	X3	2	2	3	12
Box type 4	X4	3	2	1	6
Box type 5	X5	3	2	2	12
Box type 6	X6	3	2	3	18
Box type 7	X7	4	2	1	8
Box type 8	X8	4	2	2	16
Box type 9	X9	4	2	3	24
Box type 10	X10	2	3	1	6
Box type 11	X11	2	3	2	12
Box type 12	X12	2	3	3	18
Box type 13	X13	2	4	1	8
Box type 14	X14	2	4	2	16
Box type 15	X15	2	4	3	24
Box type 16	X16	1	2	2	4
Box type 17	X17	3	2	2	12
Box type 18	X18	1	2	3	6
Box type 19	X19	2	2	3	12
Box type 20	X20	1	2	4	8
Box type 21	X21	2	2	4	16
Box type 22	X22	3	2	4	24
Box type 23	X23	1	3	2	6
Box type 24	X24	3	3	2	18
Box type 25	X25	1	4	2	8
Box type 26	X26	3	4	2	24
Box type 27	X27	2	1	2	4
Box type 28	X28	2	3	2	12
Box type 29	X29	3	1	2	6
Box type 30	X30	4	1	2	8
Box type 31	X31	4	3	2	24
Box type 32	X32	2	1	3	6
Box type 33	X33	2	1	4	8
Box type 34	X34	2	3	4	2

It is important to note that if a box has its length , width and height equal, its rotation about any of X, Y and Z axes is not considered. Similarly, if a box has it length and width equal then its rotation about Z axis is not considered. Similarly for a box which has its length and height equal no rotation about Y axis, and for a box with its width and height equal no rotation about X axis, is considered. Hence:

If l_i=w_i , no rotation about Z axis, If $l_i=h_i$, no rotation about Y axis,

If $w_i=h_i$, no rotation about X axis,

For the case where the boxes are allowed to rotate about all the axes, if the length, width and height of a box are not equal, a box can be load on the pallet in six different orientations. Hence, the following box types are equal:

$X_1=X_{16}=X_{27}$, X_2, $X_3=X_{17}=X_{28}$, $X_4=X_{10}=X_{18}=X_{23}=X_{29}=X_{32}$, $X_5=X_{11}=X_{19}$, $X_6=X_{12}=X_{24}$, $X_7=X_{13}=X_{20}=X_{25}=X_{30}=X_{33}$, $X_8=X_{14}=X_{21}$, $X_9=X_{15}=X_{22}=X_{26}=X_{31}=X_{34}$.

Let's consider the case where boxes are only allowed to rotate about Z axis, for which two orientations only are considered for each box on the pallet, if the length and width of the box are not equal. In this case the boxes in the first two column of the table 6.1.2 are the types to be worked with. These boxes are shown with their specified type in table 6.1.3, and are the box type 1 to type 15. Therefore, the following types of boxes can be shown to be equal: X_1, X_2, X_3, $X_4=X_{10}$, $X_5=X_{11}$, $X_6=X_{12}$, $X_7=X_{13}$, $X_8=X_{14}$, $X_9=X_{15}$. The rotation of the box type 1, 2, and 3 about Z axis is not considered, since they have similar length and width.

Note that no over-hanging is allowed when loading boxes on the pallet and the height of the stack on each subarea on the pallet must not exceed the palletization height. These problems are considered in the generation of strip constraints as follows:

- the length over-hanging is solved by considering that the sum of the elements in each strip type does not exceed the pallet's length,

• the width over-hanging is solved by considering that the sum of all the strip combinations times the pallet height does not exceed the pallet width times the pallet's height.

Therefore, the total area of boxes on the pallet remains less than or equal to the pallet's area.

• the stacking height is restricted to the pallet height by considering that the total volume of boxes on the pallet is less than or equal to the pallet's volume.

For this problem, there are nine types of original boxes, and they are allowed to rotate about the Z axis, and the availability of all type of boxes is certain. The number of boxes virtually doubles except for those of equal length and width. Hence, N=9, and 2N=18 However, since three boxes have their length and widths equal (k=3), the total type of boxes to be considered for two different orientations on the pallet (rotation about Z axis) are type 1 to type 15, in table 6.1.3.

The objective function can be written as:

$$\text{Maximize} \left\{ \sum_{i=1}^{9} l_i * w_i * h_i * X_i + \sum_{i=10}^{18} \hat{a}_i * l_i * w_i * h_i * X_i \right\} \qquad (6.1.1)$$

The first set of constraints for this problem is the pallet volume constraint

$$L * W * H - \left\{ \sum_{i=1}^{9} l_i * w_i * h_i * X_i + \sum_{i=10}^{18} \hat{a}_i * l_i * w_i * h_i * X_i \right\} \geq 0 \qquad (6.1.2)$$

The second set of constraints correspond to the demand restrictions on the boxes quantity. Since the length and width of boxes type 1, 2, and 3 are equal their rotation on the pallet is not considered. Box type 4 and 10 are the same, box type 5 and 11 are the same, box type 6 and 12 are the same, similarly box type 7 and 13 are the same, box type 8 and 14 are the same, and box type 9 is the same as box type 15.

$$X1 \leq 2 \qquad (6.1.3)$$

$$X2 \leq 3 \qquad (6.1.4)$$

$$X3 \leq 2 \qquad (6.1.5)$$

$$X4 + X10 \leq 3 \qquad (6.1.6)$$

$$X5 + X11 \leq 4 \qquad (6.1.7)$$

$$X6 + X12 \leq 2 \qquad (6.1.8)$$

$$X7 + X13 \leq 3 \qquad (6.1.9)$$

$$X8 + X14 \leq 3 \qquad (6.1.10)$$

$$X9 + X15 \leq 2 \qquad (6.1.11)$$

The upper bound constraint on the number of boxes of unit length is determined; such that the maximum number of unit elements to be used in the generation of the strip combinations is restricted to the pallet's width, since the width of the pallet is chopped into the unit length strips of dimension L*1. Hence we have:

$$W - X_{2N+1} \geq 0 \qquad (6.1.12)$$

However, boxes of unit length and width are not used in the formulation of the algorithm

Using equation (5.1 7) and length set equation (5.1.6), where s={4,3,2,1}; nine possible types of strip combinations are found:

a) {4,3}

b) {4,2,1}

c) {4,1,1,1}

d) {3,3,1}

e) {3,2,2}

f) {3,2,1,1}

g) {3,1,1,1,1}

h) {2,2,2,1}

j) {2,2,1,1,1}

Let A denote the number of strips combination of type "a", B denote the number of strips combination of type "b", etc., with respect to equations (5.1.8), the sum of all possible strip types must be less than or equal to the pallet width. For the pallet width being W=4, we have the following constraint;

$$A + B + C + D + E + F + G + H + J \leq 4 \qquad (6.1\ 13)$$

The other strip constraints are determined with respect to the specific lengths defined in the box length set "s" Applying equation (5.1.9), these strip constraints can be determined as

For length 1; There is one "1" in strip types "b", "d" and "h", two "1" in strip type "f", three "1" in strip types "c" and "j", and four "1" in strip type "g". Note that the box type X_{2N+1} of unit length is not considered in this constraint. Furthermore there is not any other box with length 1, hence:

$$4B + 12C + 4D + 8F + 16G + 4H + 12J \geq 0 \tag{6.1.14}$$

This is reduced to;

$$B + 3C + D + 2F + 4G + H + 3J \geq 0 \tag{6.1.14a}$$

For length 2; There is one "2" in the strip types "b" and "f", two "2" in strip types "e" and "j", and three "2" in strip type "h". Also there is one box type X1 which has a width of 2 and height 1, one box type X2 with width of 2 and height 2, one box type X3 with width 2 and height 3, one box type X10 which has width of 3 and height 1, one box type X11 with width of 3 and height 2, one box type X12 with width of 3 and height 3, one box type X13 with width of 4 and height 1, one box type X14 with width of 4 and height 2, and one box type X15 with width of 4 and height 3. Hence we have;

$$\begin{aligned} 4B + 8E + 4F + 12H + 8J \geq 4X_1 + 8X_2 + 3X_{10} \\ + 6X_{11} + 9X_{12} + 4X_{13} + 8X_{14} + 12X_{15} \end{aligned} \tag{6.1.15}$$

For length 3; There is one "3" in the strips types "a", "e", "f", and "g", there are two "3" in strip type "d". Also there is one box type X4 which has width of 2 and height 1, one box type X5 with width of 2 and height 2, and one box type X6 with width of 2 and height 3.

$$4A + 8D + 4E + 4F + 4G \geq 2X_4 + 4X_5 + 6X_6 \qquad (6.1.16)$$

For length 4; There is one "4" in strips types "a", "b", and "c". Also there is one box type X7 which has width of 2 and height 1, one box type X8 with width of 2 and height 2, and one box type X9 with width of 2 and height3.

$$4A + 4B + 4C \geq 2X_7 + 4X_8 + 6X_9 \qquad (6.1.17)$$

Note that all the strip types and all X variables are positive integer numbers.

The formulation of ILP Model 1 for this particular problem is:

$$Maximize \left\{ \sum_{i=1}^{9} l_i * w_i * h_i * X_i + \sum_{i=10}^{18} \hat{a}_i * l_i * w_i * h_i * X_i \right\}$$

Subject TO:

$$L * W * H - \left\{ \sum_{i=1}^{9} l_i * w_i * h_i * X_i + \sum_{i=10}^{18} \hat{a}_i * l_i * w_i * h_i * X_i \right\} \geq 0$$

$$X1 \leq 2$$

$$X2 \leq 3$$

$$X3 \leq 2$$

$$X4 + X10 \leq 3$$

$$X5 + X11 \leq 4$$

$$X6 + X12 \leq 2$$

$$X7 + X13 \leq 3$$

$$X8 + X14 \leq 3$$

$$X9 + X15 \leq 2$$

$$A + B + C + D + E + F + G + H + J \leq 4$$

$$B + 3C + D + 2F + 4G + H + 3J \geq 0$$

$$4B + 8E + 4F + 12H + 8J \geq 4X_1 + 8X_2 + 3X_{10} + 6X_{11} + 9X_{12} + 4X_{13} + 8X_{14} + 12X_{15}$$

$$4A + 8D + 4E + 4F + 4G \geq 2X_4 + 4X_5 + 6X_6$$

$$4A + 4B + 4C \geq 2X_7 + 4X_8 + 6X_9$$

X_i and A, B, C,....,J are positive integers.
$\hat{a}_i$ =0, if l_i=w_i , for i=10,..., 18; $\hat{a}_i$=1, otherwise.

For this particular problem, the value of objective function is found to be 112 which is the total volume of the pallet of dimension L*W*H=7*4*4=112. This means 100% pallet volume utilization. The LINDO ILP formulation can be found in APPENDIX A. The optimal values of the decision variables are as follows, where their dimension can be found in the table (6.1.3):

X_5=4, of dimensions 3*2*2,

X_7=2, of dimensions 4*2*1, and

X_8=3, of dimensions 4*2*2.

Now that the optimal results are obtained, the loading procedure begins. Loading the optimal boxes on the pallet as was discussed in the earlier chapters depends on the priority to which box must be loaded on, and/or unloaded from the pallet, and many other logical conditions for which, for instance, the loading and unloading time is reduced, etc. The most important aspect in the loading boxes in stack is the stability of the column stacks, in both static and dynamic conditions. On the other hand to achieve higher percentage of the pallet utilization the boxes with larger base area are loaded on the pallet at first; also boxes of larger volume if any, with the same base dimension as the earlier ones, can be loaded on the top to provide a better stability in terms of sliding and toppling. Since boxes of bigger volume are heavier, they are more stable with respect to the smaller and the lighter ones. Also for the case where boxes of larger base area are loaded on the pallet, a better chance can be given to the smaller ones to be loaded on top of them to provide better or higher utilization.

The optimal physical layout of the boxes stacked on the pallet with their steps of loading is shown in figure 6.1.1.

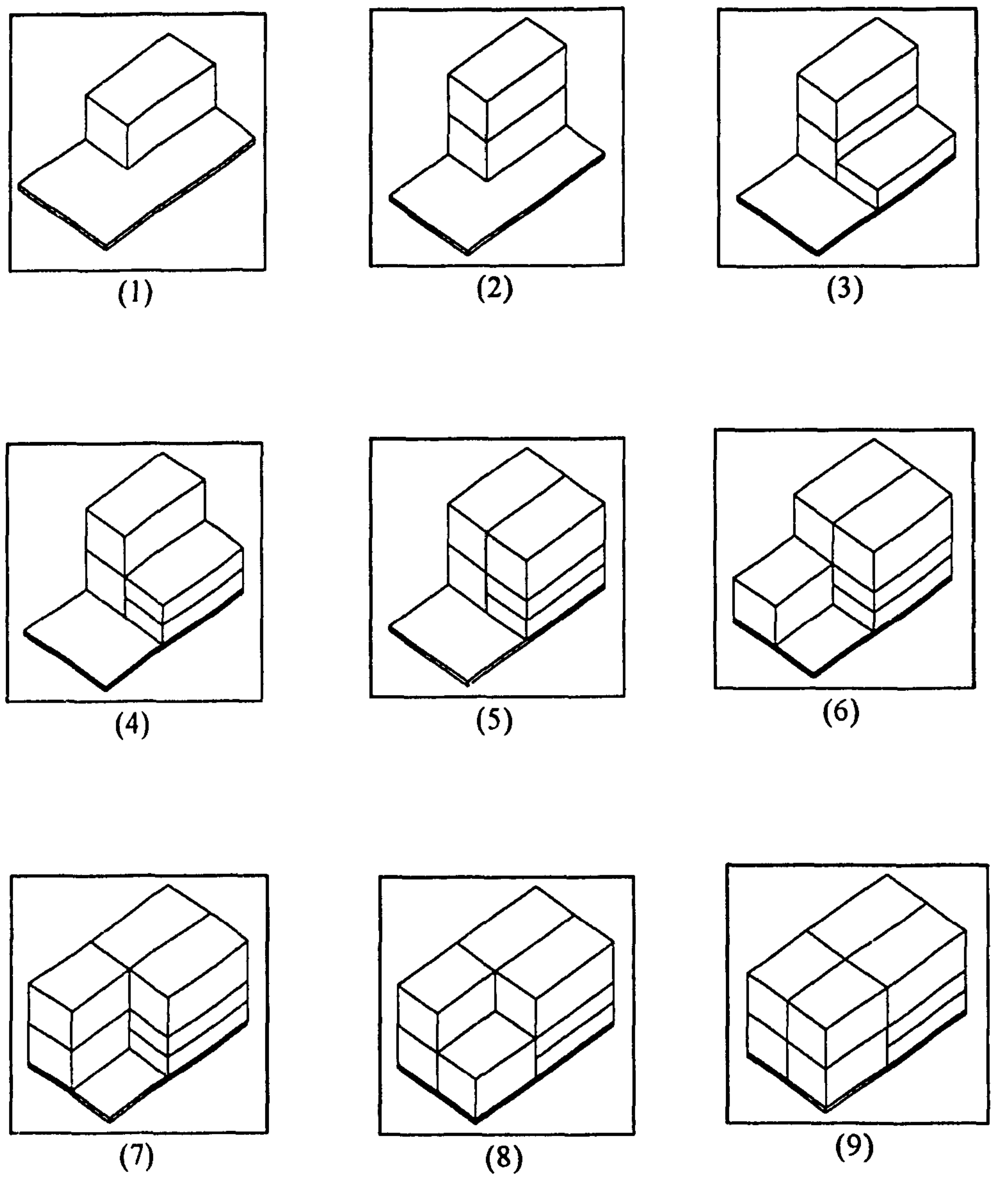


Figure 6.1.1 Optimal Physical Layout of Stacking Palletization for Model 1

6.1.1 Model 1: Case Study, with Specific Demands (Also To Be Used in 6.2 Model 2)

The solution to the Model 1 problem is obtained by employing the boxes type shown in table 6.1.3 and placing some specific demand restrictions on the number of boxes available, which are also to be used in the case study of the proposed Model 2. The number of boxes available for this specific example are:

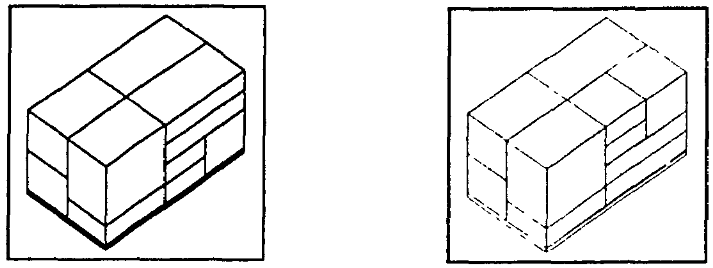
Two boxes of type 1, one box of type 2, one box of type 4, two boxes of type 5, one box of type 6, two boxes of type 7, and two boxes of type 8. However, boxes of type 3 and type 9 having height of 3 are ignored. Therefore, applying the formulation of the Model 1 problem presented in chapter 5.1.1, for volumetric optimization of the pallet, the box quantity constraints for this problem are:

$X_1 \leq 2$, $X_2 \leq 1$, $X_4 + X_{10} \leq 1$, $X_5 + X_{11} \leq 2$, $X_6 + X_{12} \leq 1$, $X_7 + X_{13} \leq 2$, $X_8 + X_{14} \leq 2$.

The optimal solution obtained for this particular problem provides a 100% volume utilization of the pallet. The value of objective function obtained is 112, and the decisions variables are:

$X_1 = 2$, $X_2 = 1$, $X_4 = 1$, $X_5 = 2$, $X_6 = 1$, $X_7 = 2$, $X_8 = 2$.

Since the volumetric solution is obtained from the proposed Model 1, boxes can sit on the pallet in many different orders and locations. The loading pattern is not restricted and depends on the preference of the operator, and the priority in which boxes to be loaded on and/or unloaded from the pallet Considering the stability of the column stacks two alternative solutions are considered. The graphical representation of the final solutions for this problem is shown in figure 6.1.2.



Solution 2 Solution 1

Figure 6.1.2 Results Obtained from the Proposed Model 1

6.1.2 Model 1: Case Study with Practical Dimensions

In the practical situations the dimension of the pallet and the boxes to be loaded on the pallet differ from those that have been used earlier in Model 1 problem The real dimensions of the pallet and the boxes used in most warehouses and industrial applications are as follows:

Pallet dimension of 48" length by 40" width, the stacking height varies, however it is usually considered as 40" height. Boxes are of various dimensions, for instance, A box of dimension 40"length, 24" width and 20" height.

The formulation of the Model 1 problem for the pallet and boxes with specified dimension above is exactly as the those mentioned earlier; however, there are slight changes in the generations of the strips combination. The box length set "s" is defined as the set in which its elements are the unique values of the length and width of the boxes The lengths/widths in this box length set are used to generate all possible types of strip combination. Therefore, the strips combination must contain the unique lengths from the box length set "s", for which the sum of their elements does not exceed the pallet length It is not necessary that the sum of the elements in the each strip type be exactly equal to the pallet length, because this may prevent the generation of some strip types

The following original types of boxes shown in table (6.1.5) are taken from Yang (1993);

Table 6.1.4 Model 1: Original Types of Boxes of Practical Dimensions

Item	Length	Width	Height	Volume	No.
Pallet	48	40	40	76800	1
Box 1	40	24	10	9600	2
Box 2	40	24	20	19200	2
Box 3	24	20	10	4800	2
Box 4	24	20	20	9600	2
Box 5	24	20	30	14400	2
Box 6	20	12	20	4800	4
Box 7	20	12	30	7200	4

The box length set "s" would be s = {40,24,20,12}

The followings are the 14 possible types of strip combination of boxes length, chosen from the elements of the box length set "s", for which the sum of their elements is less than or equal to the pallet length L=48:

a) {40}

b) {24,24}

c) {24,20}

d) {24,12,12}

e) {24,12}

f) {24}

g) {20,20}

h) {20,12,12}

i) {20,12}

j) {20}

k) {12,12,12,12}

l) {12,12,12}

m) {12,12}

n) {12}

From the strip types above it is possible to notice that, because of the dimension of the boxes, if the sum of the elements in each strip type would have been restricted to be exactly equal to the pallet length, the generation of the strip types "a", "c", "e", "f", "g", "h", "i", "j", "l", "m", and "n" would have been impossible

Using the strip types shown above, the Model 1 problem can be formulated Boxes are allowed to rotate about the Z axis, hence the number of boxes used is virtually doubled The converted box types are shown in the table 6.1.6.

Table 6.1.5 Model 1: Converted Boxes Data of Practical dimensions

Item	Length	Width	Height	Volume	Demand
Pallet	48	40	40	76800	1
Box 1	40	24	10	9600	2
Box 2	40	24	20	19200	2
Box 3	24	20	10	4800	2
Box 4	24	20	20	9600	2
Box 5	24	20	30	14400	2
Box 6	20	12	20	4800	4
Box 7	20	12	30	7200	4
Box 8	24	40	10	9600	2
Box 9	24	40	20	19200	2
Box 10	20	24	10	4800	2
Box 11	20	24	20	9600	2
Box 12	20	24	30	14400	2
Box 13	12	20	20	4800	4
Box 14	12	20	30	7200	4

The ILP formulation of this problem is given in APPENDIX B, which is in fact similar to the one shown earlier in this chapter, the optimal value of the objective function found to be 76800, which is exactly equal to the pallet volumetric dimension. This provides a 100% results of pallet volume utilization. The values of the optimal decision variables are:

X_1=2, X_9=2, X_{14}=4.

The optimal physical layout of the stacking palletization for this particular problem is shown in figure 6.1.3.

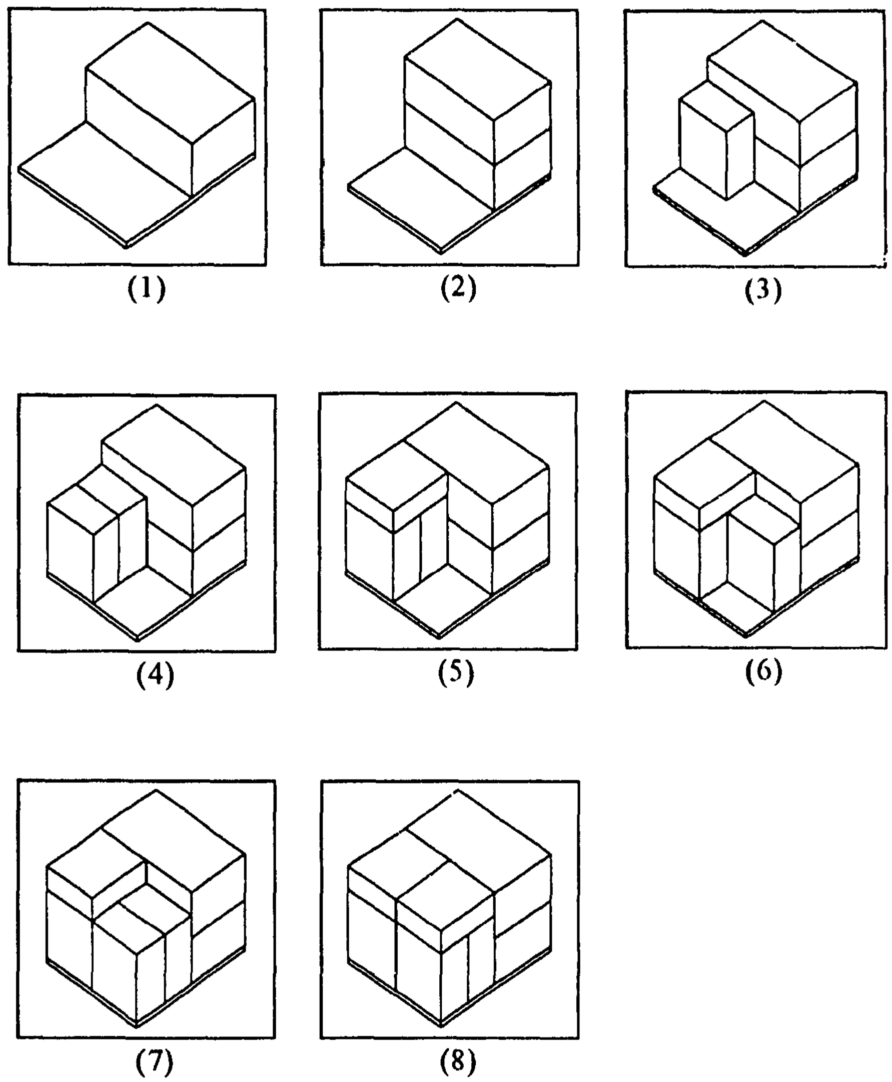


Figure 6.1.3 Optimal Physical Layout of 3D Stacking Palletization for Model 1, with Practical Dimensions.

6.2 Model 2: Case Study, using Demand Restrictions from 6.1.1 Model 1

An illustration problem is solved using the proposed ILP model 2, the demand restriction on the number of boxes of each type is considered similar to those in 6.1.1 Model 1 However, their application defers in the proposed Model 2. There are seven types of original box, the input data, of dimension l*w*h, and boxes type 3 and type 9 of height 3 are ignored, as shown in the table 6 2.1;

Table 6.2.1 Model 2: Input Data (Original Box type)

Item	Demand	Length	Width	Height	Volume
Pallet	1	7	4	4	112
Box 1	2	2	2	1	4
Box 2	1	2	2	2	8
Box 3	0	2	2	3	12
Box 4	1	3	2	1	6
Box 5	2	3	2	2	12
Box 6	1	3	2	3	18
Box 7	2	4	2	1	8
Box 8	2	4	2	2	16
Box 9	0	4	2	3	24

To determine the optimum layout on the pallet, at first the base area of the boxes are considered. Therefore, three types of original base area from these boxes can be obtained, as shown in table 6.2.2

Table 6.2.2 Model 2: Types of Boxes Base Area

Item	Number	Length	Width	Area
Pallet	1	7	4	28
Box type 1	3	2	2	4
Box type 2	4	3	2	6
Box type 3	4	4	2	8

Since boxes are considered to have two different orientations on the pallet, the number of boxes base area is virtually doubled, hence each type of box is converted into two copies, as shown in table 6.2.3.

Table 6.2.3 Model 2: Converted Boxes data

Item	Number	Length	Width	Area
Pallet	1	7	4	28
Box type 1	X1	2	2	4
Box type 2	X2	3	2	6
Box type 3	X3	4	2	8
Box type 4	X4	2	3	6
Box type 5	X5	2	4	8
Box type 6	X6	1	1	1

Note that, since box type 1 has its length and width equal, the rotation of this box is not considered on the pallet. The box type 6 is of unit length and width which is considered to ensure that at least one optimum solution can be obtained on the pallet. This is only to be considered in the constraints and it can not be included in the objective function formula. Therefore, we have the following demand on boxes type; Where X1 = 3, X2+X4 = 4; X3+X5 = 4.

At first, the objective function to determine the optimal layout on the pallet area is formulated. For this particular problem N=3, 2N=6. However, since the orientation of box type 1 is not considered, and the boxes of unit area X_6 can not be used in the formulation of the objective function, the boxes type 1 to type 5 are used.

$$\text{Maximize} \sum_{i=1}^{6} \hat{a}_i * l_i * w_i * X_i \qquad (6.2.1)$$

Furthermore, this Objective function is subjected to some specific constraints, these can be set as follows; The first set of constraints is about the pallet area

$$L*W - \sum_{i=1}^{6} \hat{a}_i * l_i * w_i * X_i \geq 0 \qquad (6.2.2)$$

The second set of constraints corresponds to the demand restrictions on the number and availability of boxes, which in fact is considered as the box quantity constraints. Considering the fact that box type 1 stands by itself, box type 2 is the same as box type 4, and box type 3 is the same as box type 5, we have:

$$X1 \leq 3 \qquad (6.2.3)$$

$$X2 + X4 \leq 4 \qquad (6.2.4)$$

$$X3 + X5 \leq 4 \qquad (6.2.5)$$

The constraint of the upper bound on the number of boxes of unit length is obtained, for which the number of boxes of unit length can not exceed the pallet width W.

$$W - X_{2N+1} \geq 0 \tag{6.2.6}$$

The strip constraints can be generated as follows:

(1) First divide the pallet of dimension 7*4 into four horizontal, unit width, strips each of dimension 7*1.

(2) Determine the box length set "s", with its unique elements s={4,3,2,1).

(3) Determine all the possible sets of strip combination, from eqation 5.1.7, such that sum of the elements in each strip set is equal to the pallet length of L=7. Nine possible strip types combination found as follows:

a) {4,3}

b) {4,2,1}

c) {4,1,1,1}

d) {3,3,1}

e) {3,2,2}

f) {3,2,1,1}

g) {3,1,1,1,1}

h) {2,2,2,1}

j) {2,2,1,1,1}

Note that strip type (i.e. {2,1,1,1,1,1}) is not a possible type of strip, since the upper bound value for the number of boxes of unit length is 4.

Let A denote the number of strips combination of type (a), B denote the number of strips combination of type (b), C denote the number of strips combination of type (c), etc. Hence with respect to equation (5.2.14), the sum of all possible strip types must not exceed the pallet width, where maximum width of the pallet is 4, the following constraint exist:

$$A + B + C + D + E + F + G + H + J \leq 4 \qquad (6.2.7)$$

Now, the strip constraints equation (5.2.15) with respect to the specific lengths can be determined:

For length 1; there is one "1" in strip types (b), (d) and (h), two "1" in strip type (f), three "1" in strip types (c) and (j), and four "1" in strip type (g), also there is only one type of box , X6, which has a width of 1.

$$B + 3C + D + 2F + 4G + H + 3J \geq X_6 \qquad (6.2.8)$$

For length 2; there is one "2" in the strip types (b) and (f), two "2" in strip types (e) and (j), and three "2" in strip type (h). Also there is one box type X1 which has a width of 2, one box type X4 which has width 3, and one box type X5 with width 4.

$$B + 2E + F + 3H + 2J \geq 2X_1 + 3X_4 + 4X_5 \qquad (6.2.9)$$

The constraints with respect to length 3 and length 4 are found in a similar manner.

For length 3; there is only one box type, type X2, which has a width of 2.

$$A + 2D + E + F + G \geq 2X_2 \qquad (6.2.10)$$

For length 4; there is only one box type, type X3, which has a width of 2.

$$A + B + C \geq 2X_3 \tag{6.2.11}$$

The ILP model for this problem, for which the optimal layout on the pallet can be determined, is formulated as shown in the following:

$$Maximize \ \sum_{i=1}^{6} \hat{a}_i * l_i * w_i * X_i$$

Subject TO:

$$L*W - \sum_{i=1}^{6} \hat{a}_i * l_i * w_i * X_i \geq 0$$

$$X1 \leq 3$$

$$X2 + X4 \leq 4$$

$$X3 + X5 \leq 4$$

$$W - X_{2N+1} \geq 0$$

$$A + B + C + D + E + F + G + H + J \leq 4$$

$$B + 3C + D + 2F + 4G + H + 3J \geq X_6$$

$$B + 2E + F + 3H + 2J \geq 2X_1 + 3X_4 + 4X_5$$

$$A + 2D + E + F + G \geq 2X_2$$

$$A + B + C \geq 2X_3$$

X_i, and A, B,...,J are positive integers.
$\hat{a}_i$=0, if l_i=w_i , for i=4,...,6; $\hat{a}_i$ =1, otherwise.

For this particular problem, the value of objective function is 28, see (APPENDIX C) This value indicates the 100% pallet area utilization. The value of the decision variables are as follows:

On the pallet area: $X_2 = 2$, $X_3 = 2$, where; X_2 has dimension (3*2), and X_3 has dimension (4*2).

The optimal physical layout of the boxes on the pallet is as shown in figure 6.2.1

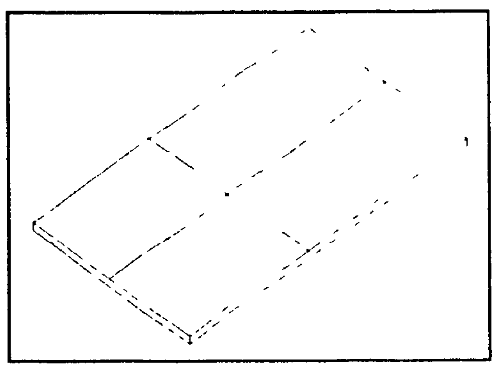


Figure 6.2.1 Optimal Physical Layout on the Pallet for Model 2.

Now, that the optimal layout on the pallet is found, the stacking height for each optimum subarea can be determined. Note that there are two different subareas, n=1,2, T=2, and each subarea repeats two times $n_1=2$, and $n_2=2$. Assign $Z_1=Z_2=X_3$, and $Z_3=Z_4=X_2$

The objective function to determine the stacking height with respect to the optimal layout on the pallet can be formulated. There are three different heights for each box, h=1,2,.. ,P, P=3. Where, $h_1=1$, $h_2=2$, and $h_3=3$.

$$\text{Maximize} \quad \sum_{j=1}^{\sum_{k=1}^{2} n_k} \sum_{h=1}^{3} h_h' * Z_{jh} \tag{6.2.12}$$

Some constraints are considered for this objective function:

The first set of constraints relates to the stacking height in each optimum subarea on the pallet

$$H - \sum_{h=1}^{3} h_h' * Z_{jh} \geq 0 \quad \forall j,\ j=1,2,\ldots,m,\ m=\sum_{k=1}^{2} n_k \tag{6.2.13}$$

The demand constraints on the boxes height quantity can be set.

$$\sum_{j=1}^{2} Z_{jh} \leq D_{1h}; \qquad \forall h \tag{6.2.14}$$

$$\sum_{j=3}^{4} Z_{jh} \leq D_{3h}; \qquad \forall h \tag{6.2.15}$$

where; $Z_{1h} = Z_{2h}$, and $Z_{3h} = Z_{4h}$ are the optimal subarea, for all heights h=1, 2, 3 and D_{11}=2, D_{12}=2, D_{13}=0, D_{31}=3, D_{32}=3, D_{33}=1 are the demand restrictions.

The ILP model for this problem, to determine the optimal stacking height in each optimum subarea, can be formulated as shown on the next page:

$$Maximize \sum_{j=1}^{\sum_{k=1}^{2} n_k} \sum_{h=1}^{3} h_h' * Z_{jh}$$

Subject to:

$$H - \sum_{h=1}^{3} h_h' * Z_{jh} \geq 0 \quad \forall j, \quad j=1,2,...,m, \quad m=\sum_{k=1}^{2} n_k$$

$$\sum_{j=1}^{2} Z_{jh} \leq D_{1h}; \qquad \forall h$$

$$\sum_{j=3}^{4} Z_{jh} \leq D_{3h}; \qquad \forall h$$

Z_{jh} are positive integer

For this particular case the optimum value of objective function is 14, (APPENDIX C) for which the stacking height in the optimum subareas Z_1 of dimension 4*2 is not totally reached. The value of decision variables are:

Z_{11}=2, two boxes of dimension 4*2*1,

Z_{22}=2, two boxes of dimension 4*2*2,

Z_{32}=2, two boxes of dimension 3*2*2,

Z_{41}=1, one box of dimension 3*2*1,

Z_{43}=1, one box of dimension 3*2*3,

The procedure is continued to find the optimum layout on the unfilled subarea of dimension 4*2, and the stacking heights are determined respectively, for which the optimum results are two boxes of dimension 2*2*1, and one box of dimension 2*2*2 (APPENDIX C).

Hence the final optimal result is 100% pallet utilization. The complete optimal physical layout of the boxes on the pallet are shown in figure 6.2.2.

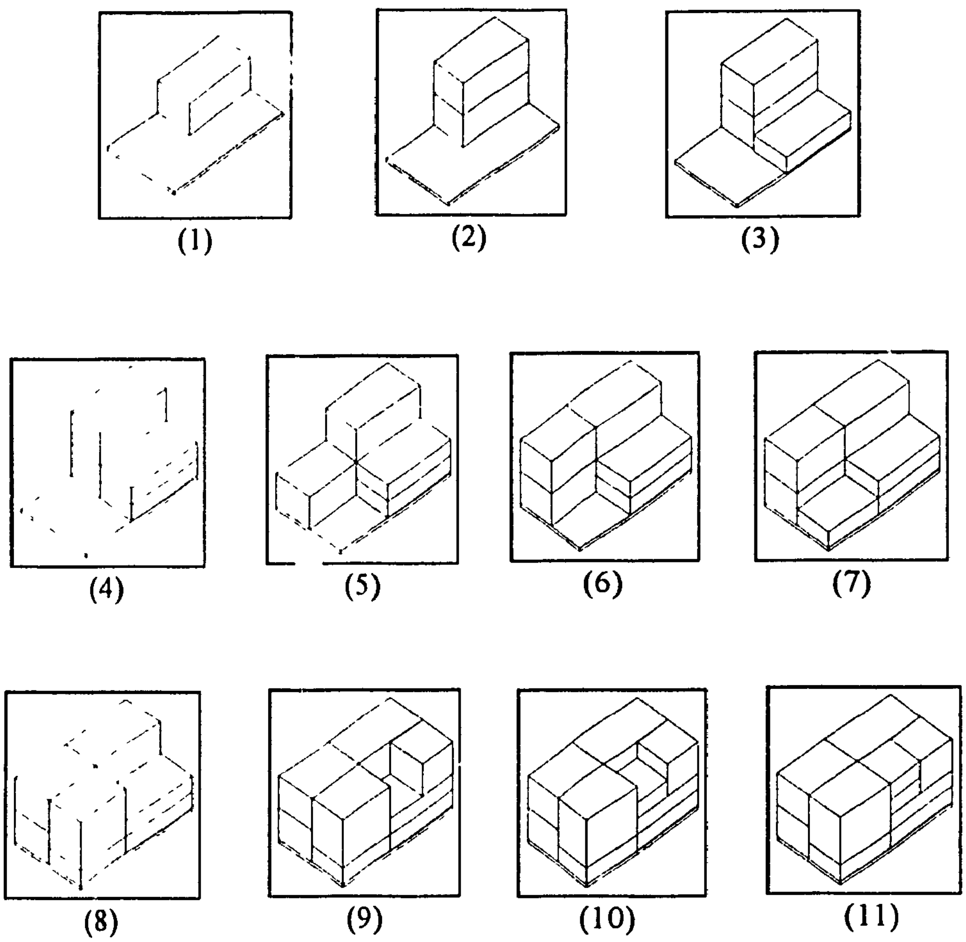


Figure 6.2.2 Optimal Physical Layout of the Stacking Palletization for Model 2.

6.2.1 Model 2: Case Study, Boxes of Practical Dimensions from 6.1.2 Model 1

The solutions to the Model 2 problem (interactive stacking palletization) is determined by using the practical dimensions and the demand restrictions on the number of available boxes used in the proposed Model 1, chapter 6.1.2 case study The procedure described in chapter 5.2.1 is employed. The number and the box types for this model are as illustrated in table 6.1.5. The solutions to the Model 2 problem for this particular case are as follows:

At first the optimal layout on the pallet of dimension 48*40 is determined Two subareas of dimension 24*40 are found. Then, for each subarea the remaining stacking height is filled. The optimal decisions values for the stacked boxes in the first subarea are. Two boxes of dimension 24*48*20. The second subarea is filled with two boxes of dimension 24*40*10; this leaves an unfilled subarea of dimension 24*40 and stacking height of 20 The optimal layout of this unfilled area is: Two subareas of dimension 24*20 and height of 20. The optimal type of boxes found are: two boxes of dimension 24*20*20. The final result reveals a 100% pallet utilization. The solution to this problem is illustrated in figure 6.2.3, on the following page.

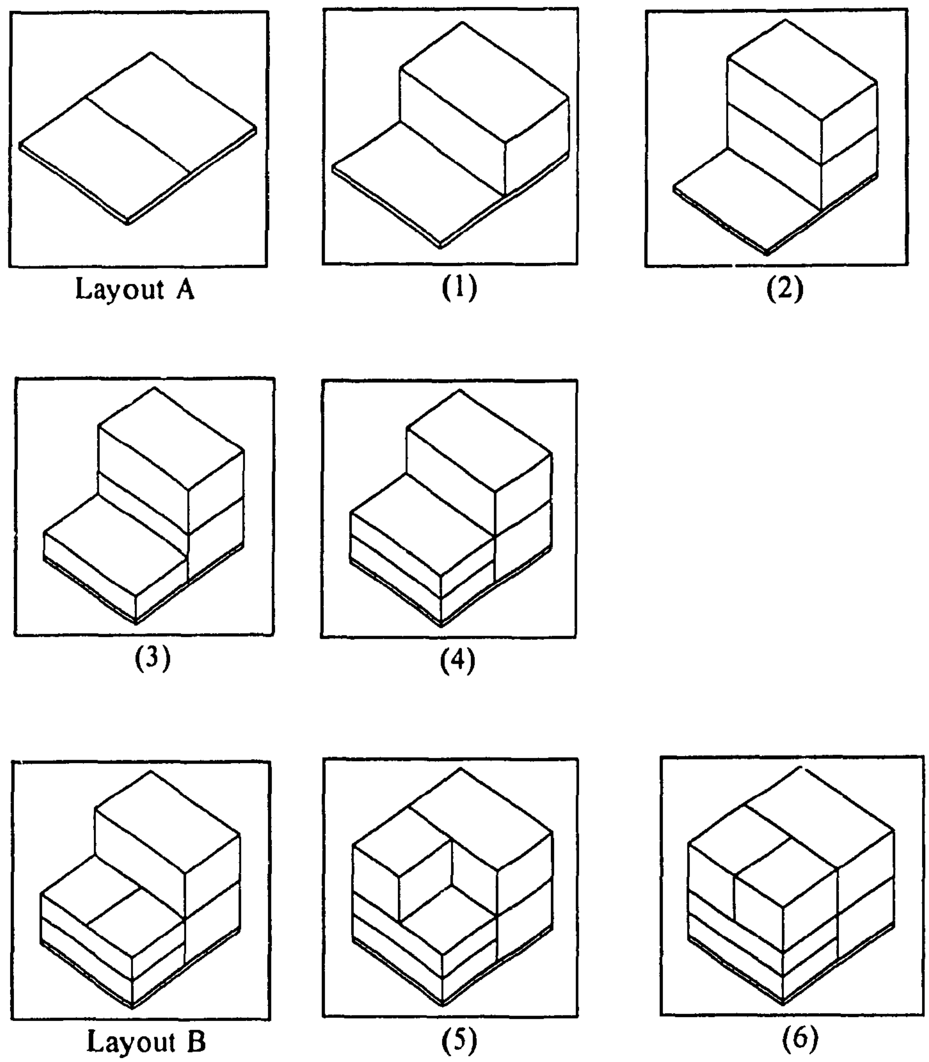


Figure 6.2.3 Results Obtained For the Proposed Model 2, Using the Number of Boxes of Practical Dimensions From Model 1.

6.3 Model 3: Case Study

The Model 3 has been developed to deal with the random pattern sequence incoming of boxes of different dimensions shown in table 6.1 5. To show the effect of random incoming sequence of boxes on the stacking palletization, three sets of simulations have been conducted for Model 3 problem. The C program developed by Yang (1993) has been modified for the proposed Model 3, so that the location of the boxes is determined and boxes are loaded on the pallet. The incoming boxes are generated by a function called ***new_box()***, which uses a random number generator to simulate the random arrival of boxes

For the case where the coming boxes are not of the required dimensions, these boxes are loaded in the WIP area and when a block of the proper size can be formed that block is moved on the column stacks on the pallet. These blocks are formed by functions called ***block20_form()*** and ***block30_form()*** The boxes used in the simulation are the same type of boxes used in the case study for Model 2 problem (see table 6.1 1) and (table 6.1.5 for boxes of practical dimensions.) The boxes loaded on the WIP area fall into three groups with respect to their height. These groups are:

Group 1 (height of 10): Type 1 and Type 3

Group 2 (height of 20): Type 2, Type 4, and Type 6

Group 3 (height of 30): Type 5, and Type 7

A new layer will be selected to fill after the box type and/or the block is selected; the height of the selected box/block assigns a height to the new layer; this is done by the function called ***new_layer()*** This new layer is then filled by selected boxes with respect to their heights, the function to select a box to load is called ***select_box i()***, where "i" indicates the box of type "i" with respect to the group it belongs to. Several other functions to load and unload boxes to and from the pallet, and WIP are used throughout the program as well (see Appendix D for the C programming language).

Three sets of simulation have been conducted with different inputs:

* **Simulation 1:** the same input data used by Yang (1993); that is, two Type 1 boxes, one Type 2 box, and two Type 5 boxes. However with different sequences than his.

* **Simulation 2:** the same input data used by Yang (1993), and with the same sequences as his.

* **Simulation 3:** the output from the example of practical dimensions for Model 2, section 6 2.1; that is, two Type 1 boxes, two Type 2 boxes, and two type 4 boxes.

Each simulation of Model 3 is run for ten different sequences. The comparison between each sequence is made by comparing Pallet Utilization, WIP, and Palletization Time. Pallet utilization is measured using formula (4.3) given in chapter 4. Maximum number of boxes in the holding area is taken as the index for WIP. The palletization time is calculated by using formula (4.4) given in chapter 4.

$$T = T_{cp} * RCP + T_{ch} * RCH + T_{hp} * RHP + T_{ph} * RPH$$

The robot arm moves at speed of 4 inches per second, therefore, each motion takes about 40 seconds. Hence the palletization time can be calculated by formula (6.3 1)

$$T = 40 * (RCP + RCH + RHP + RPH) \tag{6.3.1}$$

The output of the three simulations are shown in table 6.3.1, table 6.3.3, table 6.3.5, and the robot movement frequency and the palletization time are shown in table 6 3.2, table 6.3.4, table 6.3.6, correspondingly.

Table 6.3.1 Simulation 1: Summary of the Output

No.	Sequence →→	Time (Second)	Max. WIP area	Utilization
1	3-3-5-5-1-1-2	280	0	100%
2	3-3-2-5-1-2-5-1	360	960	100%
3	3-2-5-1-3-1-2-5	520	1920	100%
4	2-5-2-5-3-3	280	2400	100%
5	2-5-5-2-3-3	320	960	100%
6	2-5-2-5-1	240	480	100%
7	2-5-1-1-2-5-3-3	360	960	100%
8	2-2-3-1-5-3-1-5	320	960	100%
9	3-5-3-2-5-2	320	960	100%
10	3-2-3-5-5-1-2-1	440	960	100%

Table 6.3.2 Simulation 1: Robot Movement Frequency and Palletization Time

Sequence No.	RCP	RCH	RHP	RPH	Time(second)
1	7	0	0	0	280
2	6	2	1	0	360
3	4	4	4	1	520
4	5	1	1	0	280
5	4	2	2	0	320
6	4	1	1	0	240
7	6	2	1	0	360
8	5	3	0	0	320
9	4	2	2	0	320
10	4	4	3	0	440

The final stacking layout for each sequence (1 to 10) of simulation 1 is illustrated in figure 6.3.1 below.

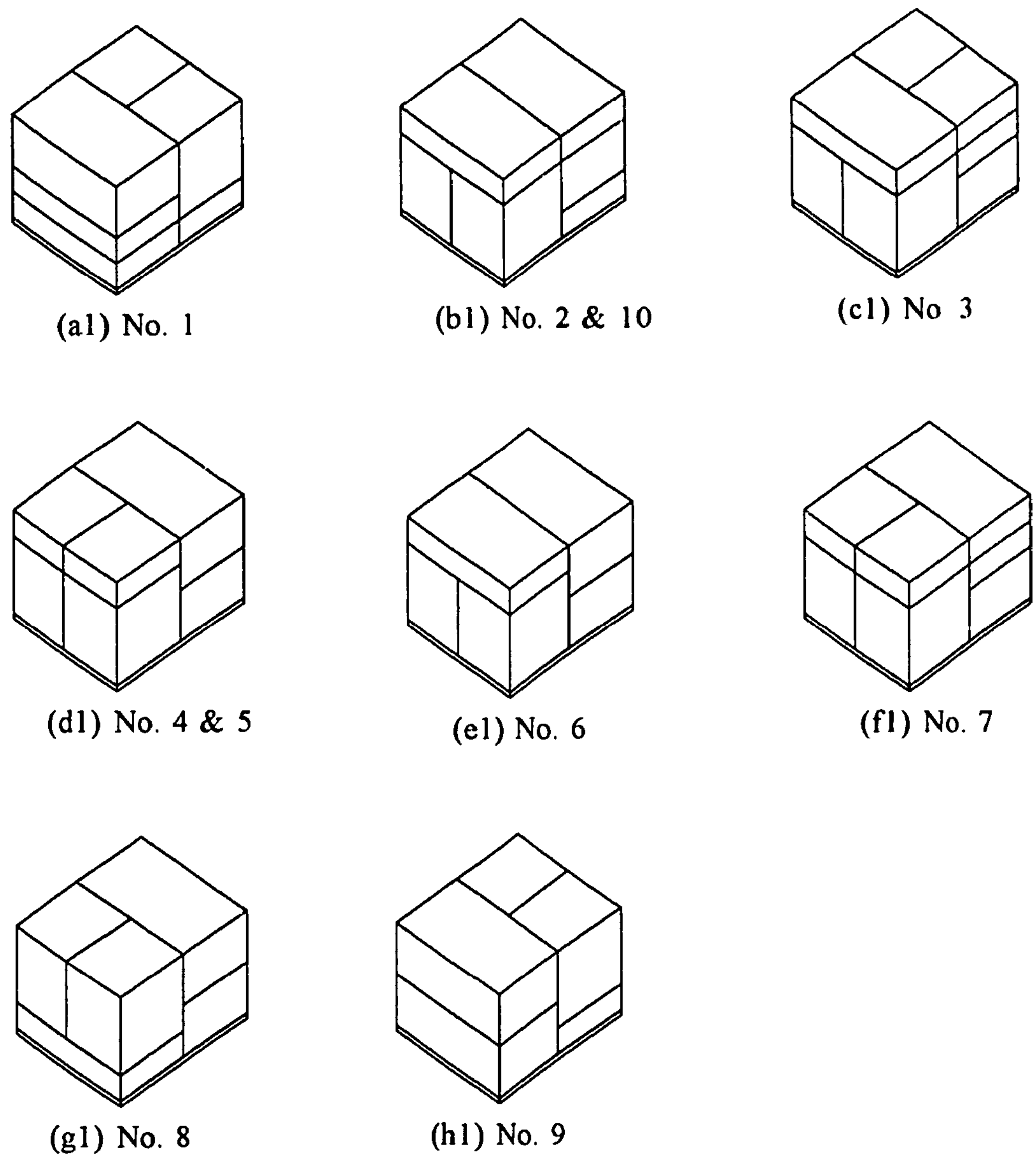


Figure 6.3.1 Stacking Layouts Obtained from Simulation 1

Table 6.3.3 Simulation 2: Summary of the Output

No.	Sequence →→	Time (Second)	Max. WIP area	Utilization
1	1-3-2-3-5-1-5	360 < (440)	960 < (2400)	100% (100%)
2	5-5-3-3-2-1-1	280 < (360)	0 < (1440)	100% (100%)
3	5-5-1-3-1-2-3	360 < (400)	1920 < (2400)	100% (100%)
4	5-5-1-3-3-1-2	280 < (400)	0 < (1920)	100% > (75%)
5	1-3-1-2-5-5-3	320 < (400)	480 < (1920)	100% (100%)
6	1-5-1-2-5-3-3	360 < (360)	1920 = (1920)	100% (100%)
7	2-3-5-5-1-3-1	440 > (400)	1440 < (1920)	100% (100%)
8	3-3-2-5-1-5-1	320 < (440)	480 < (1920)	100% (100%)
9	3-3-2-1-1-5-5	280 < (400)	0 < (1920)	100% (100%)
10	3-3-5-1-1-2-5	440 > (400)	1920 = (1920)	100% > (75%)

Table 6.3.4 Simulation 2: Robot Movement Frequency and Palletization Time

Sequence No.	RCP	RCH	RHP	RPH	Time(second)
1	5 (3)	2 (4)	2 (4)	0 (0)	360 (440)
2	7 (5)	0 (2)	0 (2)	0 (0)	280 (360)
3	5 (4)	2 (3)	2 (3)	0 (0)	360 (400)
4	7 (3)	0 (4)	0 (3)	0 (0)	280 (400)
5	6 (3)	1 (4)	1 (3)	0 (0)	320 (400)
6	5 (5)	2 (2)	2 (2)	0 (0)	360 (360)
7	5 (4)	2 (3)	3 (3)	1 (0)	440 (400)
8	6 (3)	1 (4)	1 (4)	0 (0)	320 (440)
9	7 (4)	0 (3)	0 (3)	0 (0)	280 (400)
10	5 (3)	2 (4)	3 (3)	1 (0)	440 (400)

The results in the parentheses "( )" in tables 6.3.3, and 6.3.4 are obtained by Yang (1993) for the same sequences.

The final stacking layout for each sequence (1 to 10) of simulation 2 is illustrated in figure 6.3.2.

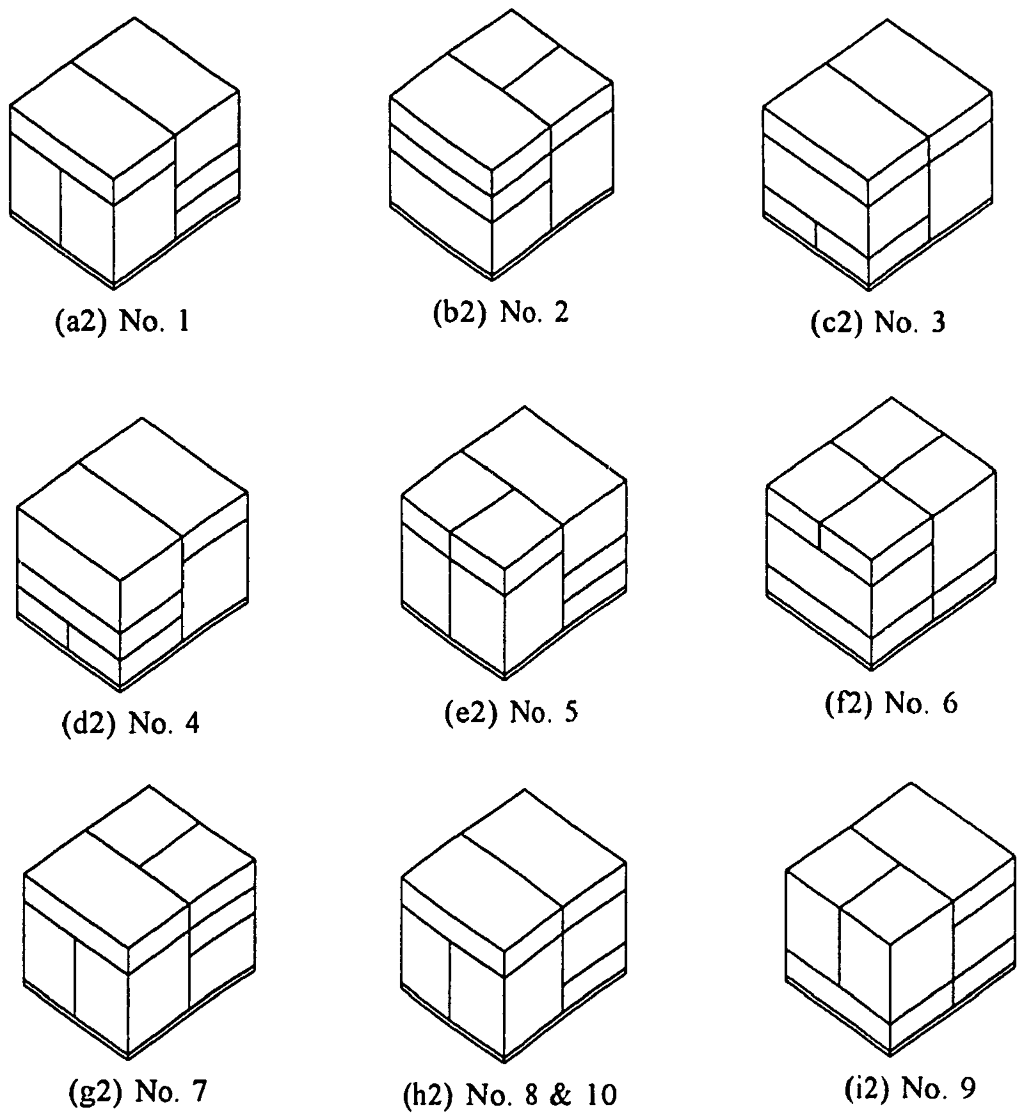


Figure 6.3.2 Stacking Layouts Obtained from Simulation 2

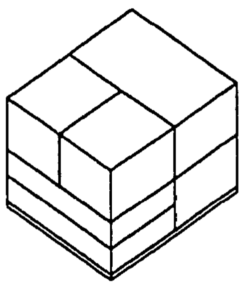
Table 6.3.5 Simulation 3: Summary of the Output

No.	Sequence →→	Time (Second)	Max. WIP area	Utilization
1	2-2-1-4-1-4	280	480	100%
2	2-2-4-4-1-1	240	0	100%
3	2-1-4-2-1-4	320	1440	100%
4	1-2-1-4-4-2	240	0	100%
5	1-4-4-1-2-2	240	0	100%
6	1-4-2-2-4-1	320	1920	100%
7	4-1-1-4-2-2	320	960	100%
8	4-2-2-1-4-1	440	2400	100%
9	2-1-4-4-1-2	320	960	100%
10	4-1-1-2-2-4	400	1440	100%

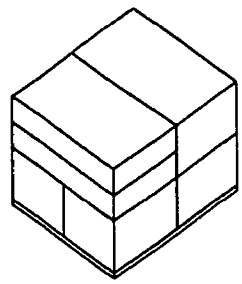
Table 6.3.6 Simulation 3: Robot Movement Frequency and Palletization Time

Sequence No.	RCP	RCH	RHP	RPH	Time{second)
1	5	1	1	0	280
2	6	0	0	0	240
3	4	2	2	0	320
4	6	0	0	0	240
5	6	0	0	0	240
6	4	2	2	0	320
7	4	2	2	0	320
8	3	3	4	1	440
9	4	2	2	0	320
10	4	2	3	1	400

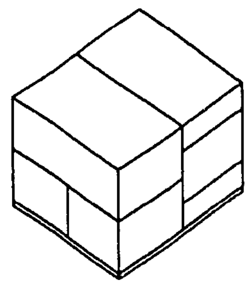
The final stacking layout for each sequence (1 to 10) of simulation 3 is illustrated in figure 6.3.3 below.



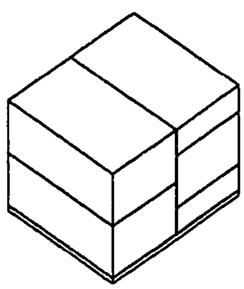
(a3) No. 1



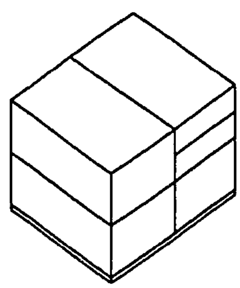
(b3) No. 2, 3 & 8



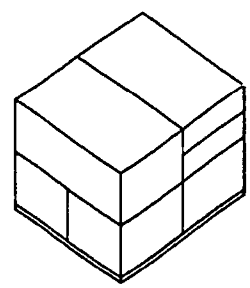
(c3) No. 4 & 6



(d3) No. 5



(e3) No. 7

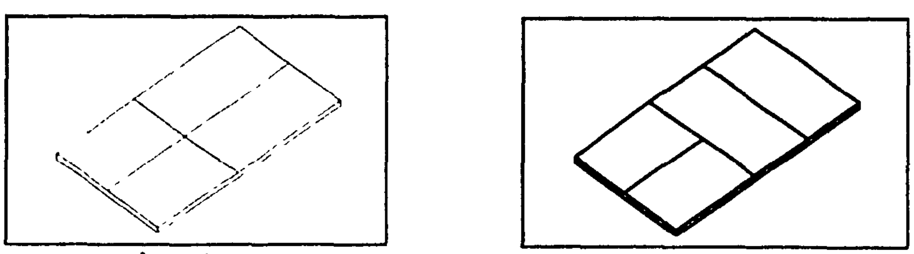


(f3) No. 9 & 10

Figure 6.3.3 Stacking Layouts Obtained from Simulation 3

CHAPTER 7. ANALYSIS OF RESULTS

Two different examples from the recent literature have been used in the research to demonstrate the efficiency and the practicality of the proposed models 1, 2 and 3. Comparison is also made between the results of the three proposed models. Tsai et al. (1988) considered a pallet of 7*4 and three types of boxes: 4*2, 3*2 and 2*2. Their method gives two optimal solutions of 2D (one layer) as illustrated in figure 7.1. The same set of data is used in the proposed Models 1 and 2. However, the third dimension is added to the boxes as well as the pallet. Therefore, in the proposed Model 1 the pallet dimension is 7*4*4, and the boxes have the same base dimension with heights of 1, 2, and 3 for each base type. In this case there are nine types of boxes, as shown in table 6.1.1. Also, another constraint, which is the number of boxes available for each type, is added. Defining the boxes type in the table 6.1.1, the number of boxes available are 2, 3, 2, 3, 4, 2, 3, 3, 2, for box Type 1 to Type 9 respectively. The optimal results obtained are: Four boxes of type 5, (3*2*2), two boxes of type 7, (4*2*1), and three boxes of type 8, (4*2*2). The volumetric result of the proposed model 1 is shown in figure 7.2.



Solution 1 Solution 2

Figure 7.1 Results Obtained from Tsai's Model

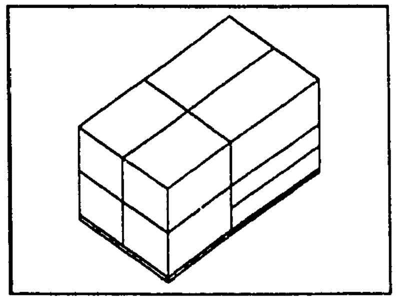


Figure 7.2 Volumetric Result Obtained from the Proposed Model 1

The pallet volumetric utilization reached by the proposed model is 100%, for this particular example. The practical difference between the solutions obtained from the proposed Model 1 compared with Tsai's solution are illustrated in table 7.1

Table 7.1 The Main Differences Between Model 1 and Tsai's Solution

Characteristics	Tsai's model	Proposed model
Pallet and Boxes of 3D	NO	YES
Volumetric Solution	NO	YES
Limits on the Number of Boxes	NO	YES

Comparing Tsai's algorithm with the proposed Model 2, the proposed Model 2 employs the same boxes as Tsai's. However, Tsai assumed that all the boxes were of the same height and he did not include these heights in his model, since he considered only one layer on the pallet, for 2D palletization. Whereas, the proposed Model 2 includes the third dimension (height) for the boxes and pallet as shown in the table 6.1.1 with some demand restrictions on the availability of boxes of each type. For this case, the stacking palletization procedure is applied. The optimal results found are: Two boxes of type 8 of dimension (4*2*2), two boxes of type 7, (4*2*1), two boxes of type 5, (3*2*2), one box of type 4, (3*2*1), and one box of type 6, (3*2*3), one box of type 2, (2*2*2), and two boxes of type 1, (2*2*1). The solution obtained for the proposed Model 2 is shown in figure 7.3.

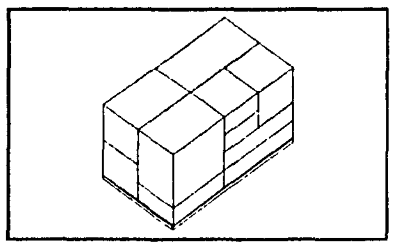


Figure 7.3 Result Obtained from the Proposed Model 2

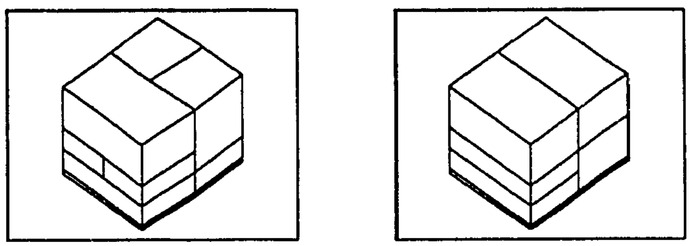
Model 2 provides an optimal solution with stacking procedure, and for this particular example the pallet volumetric utilization found to be 100%.

Table 7.2 The Main Differences Between Model 2 and Tsai's Solution

Characteristics	Tsai's model	Proposed model
Pallet and Boxes of 3D	NO	YES
Solution for Stacking	NO	YES
Volumetric Solution	NO	YES
Limits on the Number of Boxes	NO	YES

Another case discussed by Abdou and Yang (1993), they considered a pallet of dimension

48*40 and the stacking height of 40, (Dimension of 48*40*40), and boxes of practical dimensions, shown in table 6.1.5. They employed systematic procedure using the concept of layer palletization, and obtained the following optimal solutions, figure 7 4 Unlike Abdou and Yang (1993), the proposed Model 1 applies an ILP procedure to direct volumetric solutions to the pallet loading problem. The proposed Model 1 employs boxes of the same dimension as their's. However, with some different demand restrictions, it achieves the optimal result in figure 7.5.



Solution 1 Solution 2

Figure 7.4 Optimal Results Determined by Abdou and Yang

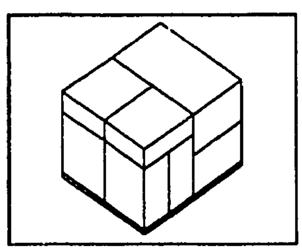


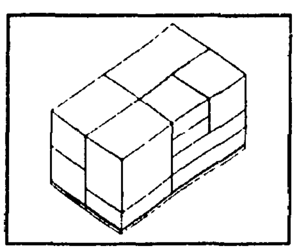
Figure 7.5 Optimal Result from the Proposed Model 1,
(with the Practical Dimensions)

The differences between two models are illustrated in the table 7.3, next page.

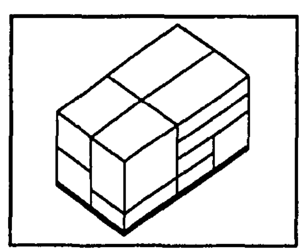
Table 7.3 The Main Differences Between Model 1 and Abdou & Yang's Model

Characteristics	Abdou & Yang's model	Proposed model
Pallet and Boxes of 3D	YES	YES
Limits on the Number of boxes	YES	YES
Solution for Stacking	NO	YES
Direct ILP Volumetric Solution	NO	YES

In another case, the comparison between the two proposed models is made. The solutions to the Model 1 and Model 2 problems are obtained by employing the same type and number of boxes available for both cases from table 6.2.1. The solutions obtained for both models are the same; however, the loading pattern in the proposed Model 1 is not restricted and depends on the preference of the operator, so that the stability and the priority of loading and/or unloading the boxes to and from the pallet can be considered.. The graphical representations of the final solutions to the Model 1 and Model 2 problems are shown in figure 7.6 and figure 7.7 respectively.



Solution 1



Solution 2

Figure 7.6 Results Obtained from the Proposed Model 1

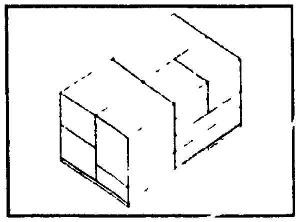
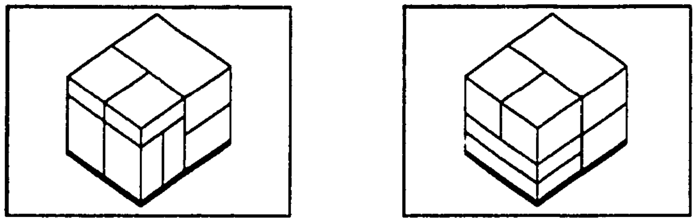


Figure 7.7 Result Obtained from the Proposed Model 2

Also, comparison is made between the two proposed Model 1 and Model 2, where boxes of practical dimensions are used with the same demand restrictions. The boxes type and number are illustrated in table 6.1.5. Comparing the results obtained from the proposed Model 2 and Model 1, it can be noted that Model 1 employs a solution to 3D volumetric optimization by mathematical procedures. In this model boxes of larger volume are selected prior to others, if they fit the pallet's upper bound dimensions. The loading layout is optional and depends on the operator's desire. The solution to the Model 1 can be either layered or stacking, depending to the boxes determined, and there are no pre-determined locations for the boxes. Whereas, Model 2 employs mathematical procedures for 3D stacking layout. At first, it determines an optimum layout (subareas) on the pallet with boxes of larger base area, and then optimizes the stacking height for each optimal subarea by selecting the boxes of proper base dimensions and tallest height, if any available and if the remaining stacking height permits In this model the location of the boxes on the pallet is pre-determined. Both models provided optimal solutions and 100% volume utilization.



Solution from Model 1 Solution from Model 2

Figure 7.8 Results from the Proposed Model 1 and 2, with Same Demand

Three simulations for the Model 3 show the effect of random incoming boxes on the palletization patterns. Simulation 1 uses the same type of boxes used by Yang (1993), however, with different sequences. The results shown in table 6.3.1 reveals that all the sequences generate 100% pallet volume utilization, for these particular examples. The loading time in that table is reasonably low since the number of RCP, shown in the column two of table 6.3.2, for each sequence is relatively high. The optimal layout of each sequence for simulation 1 is illustrated in figure 6.3.1. The stacking layouts obtained from this simulation shows that the sequence 2 and 10, and sequence 4 and 5 lay the same results. The layouts obtained for the other sequences are different from one another.

Simulation 2 uses the same type of boxes with the same sequences used by Yang (1993). The results obtained from this simulation, table 6.3.3 and 6.3.4, show that all the sequences reach 100% volume utilization, for this particular example. Whereas, the results obtained by Yang for the sequence number 4 and 10 only reach 75% utilization. The results obtained from the proposed Model 3 show a significant improvement in palletization time and WIP area. The palletization time in all the sequences, except sequence 7 and 10, are lower than those obtained by Yang. Similarly, the work-in-process (WIP) area is much lower than his result in all the sequences, such that the WIP in sequence number 2, 4, and 9 is zero. Note that, similar to the simulation 1, layouts for the sequences are different from each other except the sequence number 8 and 10 that give the same layout.

Simulation 3 uses three types of boxes obtained from the output of Model 2 problem The results for this simulation, table 6.3.5 and 6.3.6, reveals that all the sequences give 100% utilization. The palletization time is much lower than those sequences in simulation 1 and simulation 2, since the type of boxes are different. The WIP area remains low whereas, the WIP in the sequence number 2, 4, and 5 is zero. Some sequences in this simulation result in the same layout, such that the layouts obtained from sequence number 2, 3, and 8 are the same; also the layouts obtained from the sequence number 4 and 6, and the results of sequence 9 and 10 are the same. The stacking procedure in this model is base on loading boxes onto a column stack at the time, and the next column stack starts loading as soon as the previous one is completely filled A lower stacking palletization time and lower WIP can be obtained if several columns can stack up at the same time, such that if a box is not of the required type at the moment it can be loaded in the next subarea and so forth.

Comparing the results obtained from the case study of Model 2 and Model 3, one significant difference is that the layout obtained mathematically from the Model 2 problem is always the same for the same type and number of boxes. Whereas, the layout in Model 3 differs for different sequences. In the 10 sequences shown in table 6.3.5 only the sequence number 1 gives the same layout obtained from Model 2. Theoretically, Model 2 is suitable for the situation to deal with the 3D stacking palletization problem mathematically. Whereas, from the practical point of view, Model 3 is more suitable for the robotic palletization systems (RPS).

CHAPTER 8. CONCLUSIONS AND FUTURE STUDIES

The thesis has focused on maximization of pallet utilization, reduction of work-in-process (WIP) area, and reduction in pallet loading time most important criteria in the manufacturer and distribution centres palletization objectives. Two common approaches, mathematical and heuristics have been worked with. Each approach has its particular advantages: since distribution centres deal with a known number and types of boxes, the mathematical approaches are used in the development of algorithms to obtain solutions to the palletization problem. On the other hand, the mathematical approaches do not always give feasible results if all the required data, about the number and types of boxes and the pallet dimensions, are not known. This can be referred to the busy and fully automated manufacturing systems with multiple production line where the stacking palletization takes place at the end of production line with boxes of multiple size, and for which their incoming sequence is in random. Therefore, the heuristic approaches are used to deal with the solutions to such a palletization problem.

Three models are developed in this research. Model 1 provides an incitement to the mathematical ILP programming procedures for the 3D pallet volumetric optimization. The result shows that the model is capable of optimizing the pallet volume utilization to 100% with boxes of different dimensions. The result obtained can lay either in layer or stacking palletization pattern. Model 2 provides a solution for the 3D stacking palletization problem by combination of 2D palletization and stacking multiple-layer procedure. The result obtained from Model 2 is optimal. Model 3 is developed to show the effect of random sequence incoming of boxes of multi-size on the stacking pattern. The case study shows that Model 3 gives 100% utilization in all the sequences, in all three simulations, for these particular examples with the assigned multi-size boxes. Model 3 is based on a heuristic procedures and it is suitable for the automated robotic palletization in industries. The heuristic column stacking procedures for Model 3 show a great improvement in the palletization time and maintain lower WIP with respect to the heuristic model used by Yang (1993) for the layer palletization. However, since interlocking can be done in the multi-layer palletization, the stability is higher than column stacks with the guillotine cut pattern.

This study extended the research horizon from one and two-dimensional cutting to a 3D column stacking palletization and developed a mathematical ILP programming procedures for the pallet volumetric optimization. Furthermore, it extended the layer palletization to the 3D column stacking palletization, considering multi-layer in each column, by both mathematical and heuristic approaches, and used boxes of multi-size with rectangular and square shapes. However, the proposed models only considered a single pallet palletization.

The results obtained from the three developed models have shown many improvements compared to existing models. However, as a result of this experience, it is found that the palletization problem is more complex than it seemed to be, and it requires more research and investigation in this area for other improvements and more efficient results. The future study will focus particularly on the following topics:

1- since the heuristic developed in Model 3 considers stacking columns on the pallet for which second column can be started if the first column is completed, it is necessary to develop multi-column stacking heuristic procedures where many columns can stack up at the same time, such that the loading time and WIP can be greatly reduced.
2- develop CAD simulations for the proposed Model 2 and Model 3.
3- develop the physical robotic palletization system (RPS) for the heuristic Model 3.
4- implement multi-pallet system for 3D stacking of Model 3, such that two or more pallets can be loaded at the same time

BIBLIOGRAPHY

1993

G. Abdou and M. Yang (1993) **Systematic Procedure for the Palletization Problem.** Int. J. Prod. Res. To Appear 1993.

M Yang (1993) **Multi-layer Palletization of Multi-Size Boxes for 2½D and 3D Problems.** Master Thesis, Concordia University.

1992

G. Abdou and E. Lee (1992) **Contribution to the Development of Robotics Palletization of Multiple-Box Sizes.** J. Manuf. Syst. Vol 11, No. 3.

Harald Dyckhoff and Ute Finke (1992) **Cutting and Packing in Production and Distribution.** Physica-Verlag Heideberg, Germany.

K. A. Dowsland and W. B. Dowsland (1992) **Packing Problems.** Eur. J. Opl Res. 56, 2-14.

M. Girkar, B. Macleod and R. Moll (1992) **Performance Bound for Bottom-Left Guillotine Packing of Rectangles.** J. Opl Res. Soc. Vol. 43, No. 2, 169-175.

B. Chandra (1992) **Does Randomization Help in On-Line Bin Packing?** Info. Process. Lett. 43, 15-19.

1991

L. Scharge (1991) **LINDO An Optimization Modelling System 4Th Edition.** The Scientific Press, South San Francisco. 1991.

W. B. Dowsland (1991) **Three-Dimensional Packing---Solution Approach and Heuristic Development.** Int. J. Prod Res. Vol. 29, No. 8, 1673-1685.

C. S. Chen, S. Sarin and B. Ram (1991) **The Pallet Packing Problem for Non-Uniform Box Sizes.** Int. J. Prod. Res. Vol. 29, No. 10, 1963-1968.

F. Chauny, R. Loulou, S. Sadones and F. Soumis (1991) **A Two-Phase Heuristic for the Two-Dimensional Cutting-Stock Problem.** J. Opl Res. Soc. Vol. 42, No.1, 39-47.

1990

E. Bischoff and M. D. Marriott (1990) **A Comparative Evaluation of Heuristics for Container Loading.** Eur. J. Opl Res. 44, 267-276.

E. G. Coffman, Jr. and P. W Shor (1990) **Average-Case Analysis of Cutting and Packing in Two-Dimensions.** Eur. J. Opl Res. 44, 134-144.

J. J. Daniels and P. Ghandforoush (1990) **An Improved Algorithm for the Non-Guillotine-Constrained Cutting Stock Problem.** J Opl Res. Soc. 41, 141-149.

K. A. Dowsland (1990) **Efficient Automated Pallet Loading.** Eur J. Opl Res 44, 232-238.

H. Dyckhoff (1990) **A Typology of Cutting and Packing Problems.** Eur. J. Opl Res. 44, 145-159.

A. A. Farley (1990) **Selection of Stock plate Characteristics and Cutting Style for Two-Dimensional Cutting Stock Situations.** Eur J. Opl Res. 44, 239-246.

A. A. Farley (1990) **The Cutting Stock Problem in the Canvas Industry.** Eur. J. Opl Res. 44, 247-255.

H. Gehring, K. Menschner and M. Meyer (1990) **A Computer-Based Heuristic for Packing Pooled Shipment Containers.** Eur J. Opl Res. 44, 277-288

R. W. Haessler and F. B. talbot (1990) **Load Planning for Shipments of Low Density Products.** Eur. J. Opl Res. 44, 289-299.

J. F. Oliveira and J. S. Ferreira (1990) **An Improved Version of Wang's Algorithm for Two-Dimensional Cutting Problems.** Eur. J. Opl Res. 44, 256-266.

E. Lee (1990) **Automated Palletization of Multiple Box Sizes.** Master Thesis, University of Windsor.

1989

C. P. Han, K. Knott and P. J. Egbelu (1989) **A Heuristic Approach to the Three-Dimensional Cargo-Loading Problem.** Int. J. Prod. Res. 27, 757-774.

G. Löschau (1989) **Stability Criteria for Column Stocks.** Packaging Technology and Science, 2, 155-163.

1988

A. A. Farley (1988) **Practical Adaptations of the Gilmor -l 'omory Approach to Cutting Stock Problems.** Opns Res. Spektrum 10, 113-123.

A. A. Farley (1988) **Mathematical Programming Models for Cutting-Stock Problems in the Clothing Industry.** J. Opl Res. Soc. 39, 41-53.

O B. G Madsen (1988) **An Application of Travelling-Salesman Routines to Solve Pattern Allocation Problems in the Glass Industry.** J. Opl Res. Soc. 39, 249-256.

R. A. Penington and J. M A Tanchoco (1988) **Robotic Palletization of Multiple Box Sizes.** Int J Prod Res. 26, 95-105.

D W Pentico (1988) **The Discrete Two-Dimensional Assortment Problem.** Opns Res. 36, 324-332.

W. T. Rhee (1988) **Optimal Bin Packing with Items of Random Sizes.** Math. Opns Res. 13, 140-151.

D. Sculli and C. F Hui (1988) **Three-Dimensional Stacking of Containers.** Omega 16, 585-594.

R D. Tsai, E M. Malstrom and H. D. Meeks (1988) **The Two-Dimensional Palletizing Procedure for Warehouse Loading Operations.** IIE Trans. 20, 418-425.

1987

J O Berkey and P. Y. Wang (1987) **Two-Dimensional Finite Bin-Packing Algorithm.** J. Opl Res Soc. 38, 423-429.

F.•Chauny, R Loulou, S. Sadones and F. Soumis (1987) **A Two-Phase Heuristic for Strip Packing: Algorithm and Probabilistic Analysis.** Opns Res. Lett. 6, 25-33.

K. A. Dowsland (1987) **A Combined Data-Base and Algorithmic Approach to the Pallet-Loading Problem.** J. Opl Res. Soc. 38, 341-345.

K. A Dowsland (1987) **An Exact Algorithm for the Pallet Loading Problem.** Eur. J. Opl Res. 31, 78-84.

1986

F. M. Puls and J. M. A. Tanchoco (1986) **Robotic Implementation of Pallet Loading Patterns.** Int. J. Prod. Res. 24, 635-645.

1985

J. E. Beasley (1985) **An Exact Two-Dimensional Non-Guillotine Cutting Tree Search Procedure.** Opns Res. 33, 49-64.

J. B. Beasley (1985) **Bounds for Two-Dimensional Cutting.** J. Opl Res. Soc.36, 71-74

J. E Beasley (1985) **Algorithms for Unconstrained Two-Dimensional Guillotine Cutting.** J. Opl Res. Soc. 36, 297-306.

H. Carpenter and W. B. Dowsland (1985) **Practical Considerations of the Pallet-Loading Problem.** J. Opl Res. Soc. 36, 489-497.

K. A. Dowsland (1985) **Determining An Upper Bound for A Class of Rectangular Packing Problems.** Computer and Opns Res 12, 201-205.

K. A. Dowsland (1985) **A Graph-Theoretic Approach to A Pallet Loading Problem.** New Zealand Opl Res. 13, 77-86.

W. B. Dowsland (1985) **Two and Three-Dimensional Packing Problems and Solution Methods.** New Zealand Opl Res. 13, 1-18.

1984

K. A. Dowsland (1984) **The Three-Dimensional Pallet Chart: An Analysis of the Factors Affecting the Set of Feasible Layout for A Class of Two-Dimensional Packing Problems.** J. Opl Res. Soc. 35, 895-905.

H. J. Steudel (1984) **Generating Pallet Loading Patterns With Considerations of Item Starking on End and Side Surface.** J. Manuf. Syst. 3, 135-143.

1983

V. Chvatal (1983) **Linear Programming.** New York/San Francisco, in Particular **Chapter 13, "The Cutting-Stock Problem";** pp. 195-212.

T. J. Hodgson, D. S Huhes and L A. Martin-Vega (1983) **A Note on A Combined Approach to the Pallet Loading Problem.** IIE Trans. 15, 268-271.

P. Y. Wang (1983) **Two Algorithms for Constrained Two-Dimensional Cutting Stock Problems.** Opns Res. 31, 573-586.

1982

R. D. Armstrong, P. Sinha and A. A. Zoltners (1982) **The Multiple-Choice Nested Knapsack Model.** Mgmt Sci. 28, 34-43.

B. E. Bengtsson (1982) **Packing Rectangular Pieces: A Heuristic Approach.** The Computer J. 25, 353-357.

T. J. Hodgson (1982) **A Combined Approach to the Pallet Loading Problem.** IIE Trans. 14, 175-182.

M. A. Vonderembse and R.W. Heassler (1982) **A Mathematical-Programming Approach to Schedule Master Slab Caster in the Steel Industry** Mgmt Sci 28, 1450-1461.

1981

H. Dyckhoff (1981) **A New Linear Programming Approach to the Cutting Stock Problem.** Opns Res 29, 1092-1104.

1980

D. J. Brown (1980) **An Improved BL Lower Bound.** Info. Process. Lett. 11, No. 1, 37-39

A. Smith and P. De Cani (1980) **An Algorithm to Optimize the Layout of Boxes in Pallets.** J. Opnl Res. Soc. 31, 573-578.

R. W. Haessler (1980) **A Note on Computational Modifications to the Gilmore-Gomory Cutting Stock Algorithm.** Opns Res. 28, 1001-1005.

J. A. George and D. F. Robinson (1980) **A Heuristic for Packing Boxes into A Container.** Computers and Opns Res. 7, 147-156.

A. Albano and R. Orsini (1980) **A Heuristic Solution of the Rectangular Cutting Stock Problem.** The Computer J. 23, 338-343.

1979

M. Gardner (1979) **Mathematical Games: Some Packing Problems that Cannot Be Solved by Sitting on the Suitcase.** Scientific American 241(4), 22-26.

R. W. Haessler and M. A. Vonderembse (1979) **A Procedure for Solving the Master Slab Cutting Problem in Steel Industry.** AIIE Trans. 11, 160-165.

H. J. Steudel (1979) **Generating Pallet Loading Patterns: A Special Case of the Two-Dimensional Cutting Stock Problem.** Mgmt Sci. 25, 997-1004.

1978

A. K Chandra, D. S. Hirschberg, and C. K. Wong (1978) **Bin Packing with Geometric Constraints in Computer Network Design.** Opns Res. 26, 760-772.

E. G. Coffman, J. Y-T. Leung and D. W. Ting (1978) **Bin Packing: Maximizing the Number of Pieces Packed.** Acta Informatica 9, 263-271.

P. De Cani (1978) **A Note on the Two-Dimensional Rectangular Cutting-Stock Problem.** J. Opl Res. Soc. 29, 703-706.

R. W. Haessler (1978) **A Procedure for Solving the 1.5-Dimensional Coil Slitting Problem.** AIIE Trans. 10, 70-75.

1977

N. Christofides and C. Whitlock (1977) **An Algorithm for Two-Dimensional Cutting**

Problems. Opns Res. 25, 30-44.

1976

Y. Akeda and M. Hori (1976) **On Random Sequential Packing in Two and Three-Dimensions.** Biometrika 63, 361-366.

M. L. Chambers (1976) **The Cutting Stock Problem in the Flat Glass Industry-Selection of Stock Size.** Opl Res. 27, 949-957.

1975

R. W. Haessler (1975) **Controlling Cutting Pattern Changes in One-Dimensional Trim Problems.** Opns Res. 23, 483-493.

E. Page (1975) **A Note on A Two-Dimensional Dynamic Programming Problem.** Opl Res Q. 26, 321-324.

1972

I. Greenberg (1972) **Application of the Loading Algorithm to Balance Workloads.** AIIE Trans. 4, 337-339.

1966

P. C. Gilmore and R. E. Gomory (1966) **The Theory and Computation of Knapsack Functions.** Opns Res. 14, 1045-1074.

K. Kortanek and D. Sodaro (1966) **A Generalized Network Model for Three-Dimensional Cutting Stock Problems and New Product Analysis.** J. Indust. Eng. 17, 572-576.

1965

P. C. Gilmore and R. E. Gomory (1965) **Multistage Cutting Stock Problems of Two and More dimensions.** Opns Res. 13, 94-120.

M. L. Wolfson (1965) **Selecting the Best Lengths to Stock.** Opns Res. 13, 570-585.

1963

P. C. Gilmore and R. E. Gomory (1963) **A Linear Programming Approach to the Cutting Stock Problem-Part II.** Opns Res. 11, 863-888.

1962

S. N. N. Pandit (1962) **The Loading Problem.** Opns Res. 10, 639-643.

1961

P. C. Gilmore and R. E. Gomory (1961) **A Linear Programming Approach to the Cutting Stock Problem.** Opns Res. 9, 849-859.

APPENDIX A

LINDO Program for Formulation ILP Model 1

```
! M1_A3.DAT

!                       ROTATION ABOUT Z AXIZ
bat

 MAX      4 X1  +  8 X2 + 12 X3   + 6 X4  + 12 X5  + 18 X6 + 8 X7
      + 16 X8  + 24 X9 +  6 X10 + 12 X11 + 18 X12 +  8 X13
      + 16 X14 + 24 X15

SUBJECT TO

! VOLUME CONSTRIANT (RESTRICTIONS ON THE MAXIMUM VOLUME)

  1)      4 X1  +  8 X2 + 12 X3  +  6 X4  + 12 X5  + 18 X6 + 8 X7
      + 16 X8  + 24 X9 +  6 X10 + 12 X11 + 18 X12 +  8 X13
      + 16 X14 + 24 X15 <= 112
  2) X0 <= 4            ! Box of unit volume l0=w0=h0=1

! DEMAND RESTRICTIONS ON THE BOX QUANTITY.

  3) X1 <= 2
  4) X2 <= 3
  5) X3 <= 2
  6) X4 + X10 <= 3
  7) X5 + X11 <= 4
  8) X6 + X12 <= 2
  9) X7 + X13 <= 3
 10) X8 + X14 <= 3
 11) X9 + X15 <= 2

! STRIP CONSTRAINTS.

12) A + B + C + D + E + F + G + H + J <= 4
!  For length one
13) B + 3 C + D + 2 F + 4 G + H + 3 J >= 0
!  For length two
14) 4 B + 8 E + 4 F + 12 H + 8 J  - 4 X1 - 8 X2 - 12 X3 - 3 X10
- 6 X11 - 9 X12 - 4 X13 - 8 X14 - 12 X15 >= 0
!  For length three
15) 4 A + 8 D + 4 E + 4 F + 4 G - 2 X4  - 4 X5 - 6 X6 >= 0
!  For length four
16) 4 A + 4 B + 4 C - 2 X7 - 4 X8 - 6 X9 >= 0
17) A >= 0
18) B >= 0
```

```
19) C >= 0
20) D >= 0
21) E >= 0
22) F >= 0
23) G >= 0
24) H >= 0
25) J >= 0
END
GIN 25          ! GIN 44, indicates that the first 25 variables are integer,
bat             ! including the strips A to J.
leave           ! By convention LINDO assumes all the variables are non-negative.
```

! M1_A3.out

LP OPTIMUM FOUND AT STEP 5
OBJECTIVE VALUE = 112.000000
SET X15 TO <= 0 AT 1, BND= 112 0 TWIN= 112.0 8
SET X9 TO <= 0 AT 2, BND= 112.0 TWIN= 112.0 12

NEW INTEGER SOLUTION OF 112.000000 AT BRANCH 2 PIVOT 12

OBJECTIVE FUNCTION VALUE

1) 112.00000

VARIABLE	VALUE	REDUCED COST
X1	.000000	.000000
X2	.000000	.000000
X3	.000000	.000000
X4	.000000	.000000
X5	4.000000	.000000
X6	.000000	.000000
X7	2.000000	.000000
X8	3.000000	.000000
X9	.000000	.000000
X10	.000000	.000000
X11	.000000	.000000
X12	.000000	.000000
X13	.000000	.000000
X14	.000000	.000000
X15	.000000	.000000
X0	.000000	.000000
A	4.000000	.000000
B	.000000	.000000
C	.000000	.000000
D	.000000	.000000
E	.000000	.000000
F	.000000	.000000
G	.000000	.000000
H	.000000	.000000
J	.000000	.000000

ROW	SLACK OR SURPLUS	DUAL PRICES
2)	.000000	1.000000

3)	4.000000	.000000
4)	2.000000	.000000
5)	3.000000	.000000
6)	2.000000	.000000
7)	3.000000	.000000
8)	.000000	.000000
9)	2.000000	.000000
10)	1.000000	.000000
11)	.000000	.000000
12)	2.000000	.000000
13)	.000000	.000000
14)	.000000	.000000
15)	.000000	.000000
16)	.000000	.000000
17)	.000000	.000000
18)	4.000000	.000000
19)	.000000	.000000
20)	.000000	.000000
21)	.000000	.000000
22)	.000000	.000000
23)	.000000	.000000
24)	.000000	.000000
25)	.000000	.000000
26)	.000000	.000000

NO. ITERATIONS= 12
BRANCHES= 2 DETERM.= 1.000E 0
BOUND ON OPTIMUM: 112.0000
DELETE X9 AT LEVEL 2
DELETE X15 AT LEVEL 1
ENUMERATION COMPLETE. BRANCHES= 2 PIVOTS= 12

LAST INTEGER SOLUTION IS THE BEST FOUND
RE-INSTALLING BEST SOLUTION...

APPENDIX B

LINDO Program for Formulation ILP Model 1, with Practical Dimensions

```
! MMM.DAT                         Rotation About Z Axis
BAT
MAX  9600 X1  + 19200 X2  + 4800 X3  +  9600 X4 + 14400 X5
  + 4800 X6  +  7200 X7  + 9600 X8  + 19200 X9 +  4800 X10
  + 9600 X11 + 14400 X12 + 4800 X13 +  7200 X14

SUBJECT TO

 1)     9600 X1  + 19200 X2  + 4800 X3  + 9600 X4 + 14400 X5
      + 4800 X6  +  7200 X7  + 9600 X8 + 19200 X9 +  4800 X10
      + 9600 X11 + 14400 X12 + 4800 X13 + 7200 X14 <= 76800
! DEMAND RESTRICTIONS ON THE BOXES QUANTITY
 2) X1 + X8  <= 2
 3) X2 + X9  <= 2
 4) X3 + X10 <= 2
 5) X4 + X11 <= 2
 6) X5 + X12 <= 2
 7) X6 + X13 <= 4
 8) X7 + X14 <= 4
! STRIP CONSTRAINTS
 9) A + B + C + D + E + F + G + H + I + J + K + L + M + N <= 40
10) 80 D + 40 E + 80 H + 40 I + 160 K + 120 L + 80 M + 40 N
- 400 X13 - 600 X14 >= 0
11) 40 C + 80 G + 40 H + 40 I + 40 J - 240 X6 - 360 X7 - 240 X10
- 4800 X11 - 720 X12 >= 0
12) 80 B + 40 C + 40 D + 40 E + 40 F - 400 X4 - 600 X5 - 400 X8
- 800 X9 >= 0
13) 40 A - 240 X1 - 480 X2 >= 0
14) A >= 0
15) B >= 0
16) C >= 0
17) D >= 0
18) E >= 0
19) F >= 0
20) G >= 0
21) H >= 0
22) I >= 0
23) J >= 0
24) K >= 0
25) L >= 0
26) M >= 0
27) N >= 0
END
GIN 28, BAT, LEAVE
```

! MMM.OUT

LP OPTIMUM FOUND AT STEP 7
OBJECTIVE VALUE = 76800.0000
ENUMERATION COMPLETE. BRANCHES= 0 PIVOTS= 7

LAST INTEGER SOLUTION IS THE BEST FOUND
RE-INSTALLING BEST SOLUTION...

OBJECTIVE FUNCTION VALUE

1) 76800.000

VARIABLE	VALUE	REDUCED COST
X1	.000000	-9600.000000
X2	.000000	-19200.000000
X3	2.000000	-4800.000000
X4	.000000	-9600.000000
X5	.000000	-14400.000000
X6	.000000	-4800.000000
X7	.000000	-7200.000000
X8	.000000	-9600.000000
X9	2.000000	-19200.000000
X10	.000000	-4800.000000
X11	.000000	-9600.000000
X12	.000000	-14400.000000
X13	.000000	-4800.000000
X14	4.000000	-7200.000000
A	.000000	.000000
B	20.000000	.000000
C	.000000	.000000
D	.000000	.000000
E	.000000	.000000
F	.000000	.000000
G	.000000	.000000
H	.000000	.000000
I	.000000	.000000
J	.000000	.000000
K	15.000000	.000000
L	.000000	.000000
M	.000000	.000000
N	.000000	.000000

ROW	SLACK OR SURPLUS	DUAL PRICES
2)	.000000	.000000
3)	2.000000	.000000
4)	.000000	.000000
5)	.000000	.000000
6)	2.000000	.000000
7)	2.000000	.000000
8)	4.000000	.000000
9)	.000000	.000000
10)	5.000000	.000000
11)	.000000	.000000
12)	.000000	.000000
13)	.000000	.000000
14)	.000000	.000000
15)	.000000	.000000
16)	20.000000	.000000
17)	.000000	.000000
18)	.000000	.000000
19)	.000000	.000000
20)	.000000	.000000
21)	.000000	.000000
22)	.000000	.000000
23)	.000000	.000000
24)	.000000	.000000
25)	15.000000	.000000
26)	.000000	.000000
27)	.000000	.000000
28)	.000000	.000000

NO. ITERATIONS= 7
BRANCHES= 0 DETERM.= 1.000E 0

APPENDIX C

LINDO Program for formulation ILP Model 2

```
! M2_A DAT

! Determining the optimum layout for the pallet of dimension L*W=7*4.

bat
MAX  4 X1 + 6 X2 + 8 X3 + 6 X4 + 8 X5

SUBJECT TO

  1) 4 X1 + 6 X2 + 8 X3 + 6 X4 + 8 X5 <= 28
 ! Box quantity constraints, demand restrictions.
  2) X1 <= 3
  3) X2 + X4 <= 4
  4) X3 + X5 <= 4
  5) X6 <= 4
  6) A + B + C + D + E + F + G + H + J <= 4
  7) B + 3 C + D + 2 F + 4 G + H + 3 J - X6 >= 0
  8) B + 2 E + F + 3 H + 2 J - 2 X1 - 3 X4 - 4 X5 >= 0
  9) A + 2 D + E + F + G - 2 X2 >= 0
 10) A + B + C - 2 X3 >= 0
 11) A >= 0
 12) B >= 0
 13) C >= 0
 14) D >= 0
 15) E >= 0
 16) F >= 0
 17) G >= 0
 18) H >= 0
 19) J >= 0
END
GIN 15        ! GIN 15, indicates that the first 15 variables are integer,
bat           ! including the strips A to J.
leave         ! By convention LINDO assumes all the variables are non-negative.
```

! M2_A.OUT

LP OPTIMUM FOUND AT STEP 4
OBJECTIVE VALUE = 28.0000000
ENUMERATION COMPLETE. BRANCHES = 0 PIVOTS= 4

LAST INTEGER SOLUTION IS THE BEST FOUND
RE-INSTALLING BEST SOLUTION...

OBJECTIVE FUNCTION VALUE

1) 28.000000

VARIABLE	VALUE	REDUCED COST
X1	.000000	-4.000000
X2	2.000000	-6.000000
X3	2.000000	-8.000000
X4	.000000	-6.000000
X5	.000000	-8.000000
X6	.000000	.000000
A	4.000000	.000000
B	.000000	.000000
C	.000000	.000000
D	.000000	.000000
E	.000000	.000000
F	.000000	.000000
G	.000000	.000000
H	.000000	.000000
J	.000000	.000000

ROW	SLACK OR SURPLUS	DUAL PRICES
2)	.000000	.000000
3)	3.000000	.000000
4)	2.000000	.000000
5)	2.000000	.000000
6)	4.000000	.000000
7)	.000000	.000000
8)	.000000	.000000
9)	.000000	.000000
10)	.000000	.000000
11)	.000000	.000000
12)	4.000000	.000000
13)	.000000	.000000

14)	.000000	.000000
15)	.000000	.000000
16)	.000000	.000000
17)	.000000	.000000
18)	000000	.000000
19)	.000000	.000000
20)	.000000	.000000

NO. ITERATIONS= 5
BRANCHES= 0 DETERM.= 1.000E 0

```
! M2_B.dat

! Using the optimum results(optimum subareas) optained from M2_A.DAT
! to determin the optimum staking height.
! The following ILP can be set up with the specified demand restriction.
! note that the height in some columns remain unfille, because of the
! unsufficient boxes provided.
bat
MAX   1 Z11 + 2 Z12 + 3 Z13 + 1 Z21 + 2 Z22 + 3 Z23
   + 1 Z31 + 2 Z32 + 3 Z33 + 1 Z41 + 2 Z42 + 3 Z43

SUBJECT TO
 1)    Z11 + 2 Z12 + 3 Z13 <= 4
 2)    Z21 + 2 Z22 + 3 Z23 <= 4
 3)    Z31 + 2 Z32 + 3 Z33 <= 4
 4)    Z41 + 2 Z42 + 3 Z43 <= 4
 ! Box quantity contraints, demand restrictions.
 5)    Z11 + Z21 <= 2
 6)    Z12 + Z22 <= 2
 7)    Z13 + Z23 <= 0
 8)    Z31 + Z41 <= 1
 9)    Z32 + Z42 <= 2
10)    Z33 + Z43 <= 1
END
GIN 12              ! The first 12 variables are considered Integer.
                    ! By convention the variables are non-negative.
bat
leave
```

! M2_B.OUT

LP OPTIMUM FOUND AT STEP 8
OBJECTIVE VALUE = 14.0000000
ENUMERATION COMPLETE. BRANCHES= 0 PIVOTS= 8

LAST INTEGER SOLUTION IS THE BEST FOUND
RE-INSTALLING BEST SOLUTION...

OBJECTIVE FUNCTION VALUE

1) 14.000000

VARIABLE	VALUE	REDUCED COST
Z11	.000000	-1.000000
Z12	2.000000	-2.000000
Z13	.000000	-3.000000
Z21	2.000000	-1.000000
Z22	.000000	-2.000000
Z23	.000000	-3.000000
Z31	1.000000	-1.000000
Z32	.000000	-2.000000
Z33	1.000000	-3.000000
Z41	.000000	-1.000000
Z42	2.000000	-2.000000
Z43	.000000	-3.000000

ROW	SLACK OR SURPLUS	DUAL PRICES
2)	.000000	.000000
3)	2.000000	.000000
4)	.000000	.000000
5)	.000000	.000000
6)	.000000	.000000
7)	.000000	.000000
8)	.000000	.000000
9)	.000000	.000000
10)	.000000	.000000
11)	.000000	.000000

NO. ITERATIONS= 8
BRANCHES= 0 DETERM.= 1.000E 0

```
! M2_A1.DAT

! Determining the optimum layout for the subarea of dimension Lj*Wj=4*2
! this subarea was unfilled.
bat
 MAX  4 X1 + 6 X2 + 8 X3 + 6 X4 + 8 X5

SUBJECT TO

 1) 4 X1 + 6 X2 + 8 X3 + 6 X4 + 8 X5 <= 8
 2) X1 <= 4            ! Box quantity constraints, demand restrictions.
 3) X2 + X4 <= 2
 4) X3 + X5 <= 0
 5) X6 <= 2
 6) A +  B + C +  D +  E +  F +   G +   H + J <= 2
 7) B + 3 C + D + 2 F + 4 G +  H +  3 J -   X6 >= 0
 8) B + 2 E + F + 3 H + 2 J - 2 X1 - 3 X4 - 4 X5 >= 0
 9) A + 2 D + E +  F +  G - 2 X2 >= 0
10) A + B + C - 2 X3 >= 0
11) A >= 0
12) B >= 0
13) C >= 0
14) D >= 0
15) E >= 0
16) F >= 0
17) G >= 0
18) H >= 0
19) J >= 0
END
GIN 15         ! GIN 15, indicates that the first 15 variables are integer.
bat            ! including strips A to J.
leave          ! By convention LINDO assumes all the variables are non-negative
```

! M2_A1 OUT

LP OPTIMUM FOUND AT STEP 5
OBJECTIVE VALUE = 8.00000000
SET X2 TO <= 0 AT 1, BND= 8.000 TWIN= 8.000 8
SET H TO >= 2 AT 2, BND= 8.000 TWIN= 8.000 14

NEW INTEGER SOLUTION OF 8.00000000 AT BRANCH 2 PIVOT 14

OBJECTIVE FUNCTION VALUE

1) 8 0000000

VARIABLE	VALUE	REDUCED COST
X1	2.000000	.000000
X2	.000000	.000000
X3	.000000	.000000
X4	.000000	.000000
X5	.000000	.000000
X6	.000000	.000000
A	.000000	.000000
B	.000000	.000000
C	.000000	.000000
D	.000000	.000000
E	.000000	.000000
F	.000000	.000000
G	.000000	.000000
H	2.000000	.000000
J	.000000	.000000

ROW	SLACK OR SURPLUS	DUAL PRICES
2)	.000000	1.000000
3)	2.000000	.000000
4)	2.000000	.000000
5)	.000000	.000000
6)	2.000000	.000000
7)	.000000	.000000
8)	2.000000	.000000
9)	2.000000	.000000
10)	.000000	.000000
11)	.000000	.000000
12)	.000000	.000000

13)	.000000	.000000
14)	.000000	.000000
15)	.000000	.000000
16)	.000000	.000000
17)	.000000	.000000
18)	.000000	000000
19)	2.000000	.000000
20)	.000000	.000000

NO. ITERATIONS= 14
BRANCHES= 2 DETERM.= 1.000E 0
BOUND ON OPTIMUM: 8.000000
DELETE H AT LEVEL 2
DELETE X2 AT LEVEL 1
ENUMERATION COMPLETE. BRANCHES= 2 PIVOTS= 14

LAST INTEGER SOLUTION IS THE BEST FOUND
RE-INSTALLING BEST SOLUTION...

OBJECTIVE FUNCTION VALUE

1) 8.0000000

VARIABLE	VALUE	REDUCED COST
X1	2.000000	-4.000000
X2	.000000	-6.000000
X3	.000000	-8.000000
X4	.000000	-6.000000
X5	.000000	-8.000000
X6	.000000	.000000
A	.000000	.000000
B	.000000	.000000
C	.000000	.000000
D	.000000	.000000
E	.000000	.000000
F	.000000	.000000
G	.000000	.000000
H	2.000000	.000000
J	.000000	.000000

ROW	SLACK OR SURPLUS	DUAL PRICES
2)	.000000	.000000
3)	2.000000	.000000

4)	2.000000	.000000
5)	.000000	.000000
6)	2.000000	.000000
7)	.000000	.000000
8)	2 000000	.000000
9)	2.000000	.000000
10)	.000000	.000000
11)	.000000	.000000
12)	000000	.000000
13)	000000	.000000
14)	.000000	.000000
15)	.000000	.000000
16)	.000000	.000000
17)	.000000	.000000
18)	.000000	.000000
19)	2.000000	.000000
20)	.000000	.000000

NO. ITERATIONS= 14
BRANCHES= 2 DETERM.= 1.000E 0

```
! M2_B1.dat

! Using the optimum results(optimum subareas) optained from jtl_2A dat
! to determin the optimum staking height for the unfilled column.
! The following ILP can be set up with the specified demand restrictions
bat
MAX 1 Z11 + 2 Z12 + 1 Z21 + 2 Z22

SUBJECT TO

 ! remaing height
 1) 1 Z11 + 2 Z12 <= 2
 2) 1 Z21 + 2 Z12 <= 2
 ! Box quantity contraints, demand restrictions.
 3) Z11 + Z21 <= 2
 4) Z12 + Z22 <= 1          ! these should not exceed the the allowable heights
END
GIN 4               ! The first 12 variables are considered Integer.
                    ! By convention the variables are non-negative.
bat
leave
```

! M2_B1. OUT

LP OPTIMUM FOUND AT STEP 2
OBJECTIVE VALUE = 4.00000000
ENUMERATION COMPLETE. BRANCHES= 0 PIVOTS= 2

LAST INTEGER SOLUTION IS THE BEST FOUND
RE-INSTALLING BEST SOLUTION...

OBJECTIVE FUNCTION VALUE

1) 4.0000000

VARIABLE	VALUE	REDUCED COST
Z11	.000000	-1.000000
Z12	.000000	-2.000000
Z21	2 000000	-1.000000
Z22	1.000000	-2.000000

ROW	SLACK OR SURPLUS	DUAL PRICES
2)	2.000000	.000000
3)	.000000	.000000
4)	.000000	.000000
5)	.000000	.000000

NO. ITERATIONS= 2
BRANCHES= 0 DETERM.= 1.000E 0

APPENDIX D

C Program for Model 3

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

int check, s[20], HA[4], G[4],HL, X[4][10], Type[10], nn, mm, LN, No;
int L, W;
int WW[10]={0,24,24,0,0,0,0,0,0,0};     /* width of subareas */
int LL[10]={0,40,40,0,0,0,0,0,0,0};     /* length of subareas */

int T=2;            /* Number of the optimum subareas */
float S,U,UL;


/* MAIN PROGRAM */
/*-----------------------------------------------------------------------------------------*/

void main()
{
  int jj,i;
  void main_pallet();

   G[1]=0; G[2]=0; G[3]=0;
   check=0;
      for(i=0;i<10;i++)
        {
        X[1][i]=0;
        X[2][i]=0;
        X[3][i]=0;
        Type[i]=0;
        }

        for(jj=1; jj<=T; jj++)
          {
          printf("****** Stacking the subarea No. %d *****\n",jj);
          HL=0; UL=0.;
          L=LL[jj]; W=WW[jj];
          main_pallet();
          }
 }              /* END OF THE MAIN PROGRAME */

/*-----------------------------------------------------------------------------------------*/
              /* GLOBAL DATA FOR FOLLOWING SUBROUTINES */

int l[9]={0,40,40,24,24,24,20,20,40};
```

```
int w[9]={0,24,24,20,20,20,12,12,24};
int h[9]={0,10,20,10,20,30,20,30,30},

int N=7;
int H=40;
int HG[4]={0,10,20,30};
static int RN[20],G[4],TYPE[6],HR,x[10],y[10],ls,ws,
static int pallet[50][50],i,j,k,kk;

/*-----------------------------------------------------------------------------------------------*/

void main_pallet()
{
  /* check=0, if there is no box on the pallet.
     check=1, if there are boxes on the pallet, but the layer is not
              completed.
     check=2, if there are boxes on the pallet, a layer is completed.


    U is the pallet utilization
    HR is remaining height
    HL is the height of a selected layer
    HG[i] is the height of boxes in group i
    G[i] is the total number of boxes of type i in the system
    B[i] is number of boxes on layer i
    M is the total number of box groups
    N is the total number of box types
    n is the total number of boxes
    n[i] is the number of boxes of type i
     */
printf("\n");
printf("***** Program for 3D Stacking Heuristic Procedure *****\n");
printf("\n");

/* initailizing */
S=L*W;
No=1; LN=1; nn=1; mm=0;

for(i=1;i<=N+1;i++) s[i]=l[i]*w[i];

U=0.0; UL=0.0;
check=0;
HR=H;
```

```
randomize();


while(HR != 0)
       {
       switch(check)
         {
case 0:
          new_layer();
          check=1;
          break;
case 1:
          fill_layer();
          ++LN;
          nn=1;
          printf("The utilization of layer[%d] is %f\n",LN-1,UL);
          printf(" \n");
          UL=0.0;
          HR=HR-HL;
          printf("The remaining pallet height is %d\n",HR);
          printf(" \n");
          check=0;
          break;
          }

          G[1]=Type[1]+Type[3];
          G[2]=Type[2]+Type[4]+Type[6];
          G[3]=Type[5]+Type[7];

          printf("G[1]=%d G[2]=%d G[3]=%d\n",G[1],G[2],G[3]);
          printf(" \n");
       }
return;
}

/*---------------------------------------------------------------------------------------------------*/
                     /* Function for New Layer Selection */
new_layer()
{
switch(HR)
       {
case 10:
       HL=10;
       break;
```

```
case 20:

        if (G[2] != 0) HL=HG[2];
        else if (G[1]==0)
                {
                 new_box();
                 block20_form();
                 if(G[1] > 0) HL=HG[1];
                 else if(G[2] > 0) HL=HG[2];
                      else { while( G[1]==0  && G[2]==0) new_box(),
                              if (G[1]==1) HL=HG[1];
                              if (G[2]==1) HL=HG[2];
                             }
                 }
                 else HL=HG[1];
        break;
case 30:

        if (G[3] != 0 ) HL=HG[3];
        else if (G[2] != 0) HL=HG[2];
           else if (G[1]==0)
                 {
                 new_box();
                 block30_form();
                 if(G[3] > 0) HL=HG[3];
                 else if(G[2] > 0) HL=HG[2];
                    else if(G[1] > 0) HL=HG[1];
                 }
        else HL=HG[1];
        break;
case 40:

        if (G[3] != 0 ) HL=HG[3];
        else if (G[2] != 0) HL=HG[2];
           else if (G[1]==0)
                 {
                 new_box();
                 if(G[3] > 0) HL=HG[3];
                 else if(G[2] > 0) HL=HG[2];
                 else if(G[1] > 0) HL=HG[1];
                 }
                 else HL=HG[1];
        break;
```

```
  default.
            printf("**********NO SUCH HR**********\n");   break;
        }

        printf("The height for this layer is %d\n",HL);
        printf(" \n");
return;
}
/*------------------------------------------------------------------------------------------*/


fill_layer()
{

int size,

printf("Initialize the subareas on the pallet:\n");
printf(" \n");

for(i=1; i<=W; i++)
        {
        for(j=1, j<=L; j++)
            {
            pallet[i][j]=0;
            }
        }

    switch(HL)
    {
case 10:
        select_box1();
        break;
case 20:
        select_box2();
        break;
case 30:
        select_box3();
        break;
    }

return,
}

/*------------------------------------------------------------------------------------------*/
```

```
        /***** Function for Selecting Boxes to load *****/

select_box1()
{
  for(i=1;;i++)
    {
    if(G[1] > 0)
      {
      if(Type[1] >= 1) X[LN][nn]=1;
      else X[LN][nn]=3;
      printf(" \n");
      printf("X[%d][%d]=%d\n",LN,nn,X[LN][nn]);
      printf(" \n");

      load_box();

      if(G[1] == 1)
        {
        for(i=1; i<=8 ;i++)
           {
           if(Type[i] == 1 && h[i] == 10)
             {
             if((UL+s[i]*1.0/S) == 1)
                {
                X[LN][nn]=i;
                load_box();
                }
              break;
              }
           }
        }

      if(UL == 1) break;
      }
   else new_box();
   }

return;
}
/*-----------------------------------------------------------------------------------------------*/

select_box2()
{
```

```
        for(i=1,,i++)
        {
        if(G[2] > 0)
            {
            if(Type[2] >=1) X[LN][nn]=2;
            else if(Type[4] >= 1) X[LN][nn]=4;
            else X[LN][nn]=6;
            printf("X[%d][%d]=%d\n",LN,nn,X[LN][nn]);
            printf(" \n");

            load_box();

            if(G[2] == 1)
               {
               for(i=1; i<=8 ;i++)
                  {
                  if(Type[i] == 1 && h[i] == 20)
                     {
                     if((UL+s[i]*1.0/S) == 1)
                        {
                        X[LN][nn]=i;
                        load_box();
                        }
                     break;
                     }
                  }
               }
            if(UL == 1) break;
           }
        else new_box();
        }

return;
}
/*------------------------------------------------------------------------------------------------*/

select_box3()
{

    for(i=1;;i++)
       {
       if(G[3] > 0)
          {
          if(Type[8] >= 1) X[LN][nn]=8;
```

```
          else if(Type[5] >= 1) X[LN][nn]=5,
             else X[LN][nn]=7;
             printf("X[%d][%d]=%d\n",LN,nn,X[LN][nn]),
             printf(" \n");

             load_box();

        if(G[3] == 1)
            {
            for(i=1; i<=8 ;i++)
               {
               if(Type[i] == 1 && h[i] == 30)
                  {
                  if((UL+s[i]*1.0/S) == 1.0)
                     {
                     X[LN][nn]=i;
                     load_box();
                     }
                      break;
                  }
               }
            }

           if(UL == 1) break;
          }
        else new_box();
      }

return;
}
/*-----------------------------------------------------------------------------------------------------------*/

load_box()
{
  int k;

  for(i=1; i<=L; i++)
     {
     for(j=1; j<=W; j++)
        {
        if(pallet[j][i] == 0)
           {
           ws=0; ls=0;
```

```
for(kk=1, kk<=W; ++kk)
   {
   if(pallet[kk][i] == 0) ++ws;
   }

    for(k=1, k<=L; k++)
       {
       if(pallet[j][k] == 0) ++ls;
       }

       x[nn]=j-1; y[nn]=i-1;
       printf("ws=%d ls=%d %d %d\n", ws,ls,x[nn],y[nn]);
       printf(" \n");

       if( W%l[X[LN][nn]] == 0.0)
         {
         if(ls >= w[X[LN][nn]] && ws >= l[X[LN][nn]])
              {
if(l[X[LN][nn]] == 20 && l[X[LN][nn-1]] == 20 && x[nn-1] == 0)
        {
        x[nn]=0; y[nn]=y[nn-1]+12;
        }
        put_box(x[nn],y[nn],w[X[LN][nn]],l[X[LN][nn]]);
        ++nn;
        mm=1;
              }
          }

   else if( W%w[X[LN][nn]] == 0.0)
      {
      if(ls >= l[X[LN][nn]] && ws >= w[X[LN][nn]])
          {
          if(w[X[LN][nn]] == 20)
              {
              if(x[nn] == 20 && y[nn] == 12)
                  {
                  x[nn] = 0; y[nn] = 24;
                  }
               if(x[nn] == 0 && y[nn] == 12)
                 {
                 x[nn] = 20; y[nn] = 24;
                 }
              }
  put_box(x[nn],y[nn],l[X[LN][nn]],w[X[LN][nn]]),
```

```
++nn;
mm=1;
                }
              }
              break;
            }
          }
          if(mm==1) break;
        }
      if(mm == 0) remove_box(),
       else
          {
          printf("Type %d box is loaded\n",X[LN][nn-1]);
          printf(" \n");

          UL=UL+s[X[LN][nn-1]]*1.0/S;

          printf("The layer utilization UL=%f\n",UL);
          printf(" \n");

             for(i=1; i<=8 ,i++)
               {
               if(X[LN][nn-1] == i)
                 {
                 Type[i]=Type[i]-1;

switch(i)
       {
case 1:
       --G[1];
       break;
case 2:
       --G[2];
       break,
case 3:
       --G[1];
       break;
case 4:
       --G[2];
       break;
case 5:
       --G[3];
       break;
case 6:
```

```
        --G[2];
        break;
case 7:
        --G[3];
        break;
case 8:
        --G[3];
        break,
      }
        break,
     }
   }
 }
  printf("The boxes in the holding area are:\n");
  printf("  \n"),

  for(i=1;i<=8;i++)
    {
    if(Type[i] != 0) printf("Type[%d]=%d\n",i,Type[i]);
    }
    printf(" \n");
    printf("%d  %d  %d\n",G[1], G[2], G[3]);
    printf(" \n");

    mm=0,

return;
}
/*-----------------------------------------------------------------------------------------------*/

put_box(xx,yy,ww,ll)
{
  int a,b;

/* printf("%d %d\n",ww,ll); */

  for(i=1; i<=ll, i++)
    {
    a=xx+i;
    for(j=1; j<=ww; j++)
      {
      b=yy+j;
      pallet[a][b]=1;
      }
```

```
    }

return;
}
/*-----------------------------------------------------------------------------------------------*/

remove_box()
{
  int a,b,c,d;

  printf("nn=%d\n",nn);
  printf(" \n");

  for(i=1; i<=nn-1; i++)
     {
     printf("X[%d][%d]=%d\n",LN,i,X[LN][i]),
     printf(" \n");

     if(((s[X[LN][nn]]-s[X[LN][i]])*1 0/S+UL) == 1)
           {
           if( W%l[X[LN][i]] == 0.0)
              {
              for(a=1; a<=l[X[LN][i]]; a++)
                   {
                   for(b=1; b<=w[X[LN][i]], b++)
                      {
                      c=a+x[i];
                      d=b+y[i];
                      pallet[c][d]=0;
                      }
                   }
              }
           else if( W%w[X[LN][i]] == 0.0)
               {
               for(a=1; a<=w[X[LN][i]]; a++)
                   {
                   for(b=1; b<=l[X[LN][i]]; b++)
                      {
                      c=a+x[i];
                      d=b+y[i];
                      pallet[a][b]=0;
                      }
                   }
               }
```

```
            UL=UL-s[X[LN][i]]*1.0/S;

            printf("x[%d]=%d y[%d]=%d\n",i,x[i],i,y[i]);
            printf(" \n");

            for(k=1, k<=8 ,k++)
               {
               if(X[LN][i] == k)
                   {
                   printf("%d      ",Type[k]);
                   ++Type[k];

                   printf(" \n");
                   printf("%d    \n ",Type[k]);
                   printf(" \n");

   printf("Put type[%d] box back to the holding area\n",k);
   printf("\n");

switch(k)
        {
case 1:
        ++G[1];
        break;
case 2:
        ++G[2];
        break;
case 3:
        ++G[1];
        break;
case 4:
        ++G[2];
        break;
case 5:
        ++G[3],
        break;
case 6:
        ++G[2],
        break;
case 7:
        ++G[3];
        break;
case 8:
        ++G[3];
```

```
        break;
        }
          break;
                  }
               }
           }
     }

return;
}
/*-----------------------------------------------------------------------------------------------*/


new_box()      /* Function for New Box Generation */

{
   int ok=0;

   if(TYPE[1]+TYPE[2]+TYPE[3]+TYPE[5] == 8) exit(0);

   while(ok != 1)
     {
     RN[No]=random(5)+1;
     if(RN[No]!=4)
        {
        ++TYPE[RN[No]];
        if(TYPE[RN[No]]<=2) ok=1;
        else --TYPE[RN[No]];
        }
        else ok=0;
     }
     printf("No. %d box is a type %d box \n", No, RN[No]);
     printf(" \n");

     for(i=1;i<=7;i++)
       {
       if(RN[No]==i) ++Type[i];
       }

        if(h[RN[No]]==10) ++G[1];
        if(h[RN[No]]==20) ++G[2];
        if(h[RN[No]]==30) ++G[3];

        ++No;
```

```
        for(i=1;i<=5;i++) printf("TYPE[%d]=%d\n", i,TYPE[i]);

return;
}
/*-----------------------------------------------------------------------------------------------*/

block20_form() /* Function for Generating Blocks with height of 20 */
{

   while(Type[3] >= 2)
       {
       Type[3]=Type[3]-2;
       Type[4]=Type[4]+1;
       ++G[2];
       }

    while(Type[1] >= 2)
        {
        Type[1]=Type[1]-2;
        Type[2]=Type[2]+1;
        ++G[2];
        }

    while(Type[1] >= 1 && Type[3] >= 2)
          {
          Type[1]=Type[1]-1;
          Type[3]=Type[3]-2;
          Type[2]=Type[2]+1;
          ++G[2];
          }
return,
}
/*-----------------------------------------------------------------------------------------------*/

block30_form()      /* Function for Generating Blocks with height of 30 */

{
    while(Type[3] >=1 && Type[4]>=1)
       {
       Type[3] = Type[3]-1;
       Type[4] = Type[4]-1;
       Type[5] = Type[5]+1;
       ++G[3];
       }
```

```
while(Type[3] >=1 && Type[6]>=2)
    {
    Type[3] = Type[3]-1;
    Type[6] = Type[6]-2;
    Type[5] = Type[5]+1;
    ++G[3],
    }

while(Type[3] >=3)
     {
     Type[3] = Type[3]-3;
     Type[5] = Type[5]+1;
     ++G[3];
     }

while(Type[1] >=1 && Type[2]>=1)
      {
      Type[1] = Type[1]-1;
      Type[2] = Type[2]-1;
      Type[8] = Type[8]+1;
      }

while(Type[1] >=1 && Type[4]>=1 && Type[6]>=2)
    {
    Type[1] = Type[1]-1;
    Type[4] = Type[4]-1;
    Type[6] = Type[6]-2;
    Type[8] = Type[8]+1,
    }

while(Type[1] >=1 && Type[4]>=1 && Type[3]>=2)
      {
      Type[1] = Type[1]-1;
      Type[4] = Type[4]-1;
      Type[3] = Type[3]-2;
      Type[8] = Type[8]+1;
      }

while(Type[1] >=1 && Type[4]>=2)
       {
       Type[1] = Type[1]-1;
       Type[4] = Type[4]-2;
       Type[8] = Type[8]+1;
       }
```

```
while(Type[1] >=1 && Type[3]>=2 && Type[6]>=2)
    {
    Type[1] = Type[1]-1;
    Type[6] = Type[6]-2;
    Type[3] = Type[3]-2;
    Type[8] = Type[8]+1;
    }

while(Type[1] >=1 && Type[3]>=4)
      {
      Type[1] = Type[1]-1;
      Type[3] = Type[3]-4;
      Type[8] = Type[8]+1;
      }

while(Type[1] >=1 && Type[6]>=4)
        {
        Type[1] = Type[1]-1;
        Type[6] = Type[6]-4;
        Type[8] = Type[8]+1;
        }

while(Type[1]>=3)
  {
  Type[1] = Type[1]-3;
  Type[8] = Type[8]+1;
  }

while(Type[1] >=2 && Type[3]>=2)
    {
    Type[1] = Type[1]-2;
    Type[3] = Type[3]-2;
    Type[8] = Type[8]+1;
    }

while(Type[2] >=1 && Type[3]>=2)
        {
        Type[2] = Type[2]-1;
        Type[3] = Type[3]-2;
        Type[8] = Type[8]+1;
        }

for(i=1; ;i++)
    {
```

```
        if(Type[8]==i) G[3]=G[3]+i;
        break;
        }

return,
}
/*-----------------------------------------------------------------------------------------------------*/
/*------------------***** END OF THE SUBRUTINE PROGRAMS *****----------*/
```