AD 717276

United States Naval Postgraduate School

THESIS

AREA-EFFICIENT AND VOLUME-EFFICIENT ALGORITHMS FOR LOADING CARGO

by

Ernest Larry DeSha

September 1970

This document has been approved for public release and sale; its distribution is unlimited.

D D C
OCT 21 1970

43

Area-Efficient and Volume-Efficient Algorithms

for

Loading Cargo

by

Ernest Larry DeSha
Lieutenant Commander, United States Navy
B.S., United States Naval Academy, 1961

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
September 1970

Author Ernest Larry DeSha

Approved by: Alan W. McMasters
Thesis Advisor

W. M. Woods for J. R. Borsting
Chairman, Department of Operations Analysis

Milton U. Clauser
Academic Dean

ABSTRACT

The proposed greater reliance upon airlifting military forces demands that cargo loading time be minimized while utilization of aircraft cargo compartment space is maximized. Two loading algorithms have been developed with these goals in mind - a two dimensional one for loading cargo where all items must be placed on the floor, and a three-dimensional one for cargo which can be stacked. The three-dimensional algorithm consists of the two-dimensional algorithm and a special stacking algorithm. Tests using randomly generated three-dimensional cargo lists indicate that 90% area efficiencies for the two-dimensional and 80% volumetric efficiencies for the three-dimensional algorithm are possible. These algorithms were designed for either hand calculations or computer calculations.

TABLE OF CONTENTS

I. INTRODUCTION ---------- 9

A. BACKGROUND ---------- 9

B. THE NEED FOR BETTER LOADING ALGORITHMS ---- 10

C. RELIANCE ON HEURISTIC METHODS ---------- 13

D. A STANDARD FOR LOADING ALGORITHM COMPARISONS ---------- 14

II. OBJECTIVES AND SCOPE ---------- 15

III. THE SYNTHETIC TEST LOADS ---------- 17

IV. THE STACKING ALGORITHM ---------- 21

A. GENERAL ---------- 21

B. METHOD ---------- 22

C. TEST RESULTS ---------- 25

D. SUMMARY ---------- 27

V. THE LENGTH-MODULAR ALGORITHM ---------- 28

A. GENERAL ---------- 28

B. METHOD ---------- 28

C. TEST RESULTS ---------- 32

D. SUMMARY ---------- 36

VI. AREAS FOR FURTHER STUDY ---------- 37

A. METHODS WITH "THIS END UP" ASSUMPTION REMOVED ---------- 37

B. PRE-STACKING ---------- 37

C. HEIGHT-MODULAR STACKING ---------- 38

D WEIGHT CONSIDERATIONS ---------- 38

E. NON-RECTANGULAR CONTAINERS ---------- 38

VII. CONCLUSIONS AND RECOMMENDATIONS ---------- 39

Preceding page blank

COMPUTER PROGRAM ---------- 40

LIST OF REFERENCES ---------- 44

INITIAL DISTRIBUTION LIST ---------- 45

FORM DD 1473 ---------- 47

 Preceding page blank

LIST OF TABLES

Table

I. Sample distribution of cargo items by volume ---------- 19

II. Sample distribution of cargo items by volume ---------- 20

III. Efficiencies of the stacking algorithm ---------- 25

IV. Area efficiency comparison of the length-modular and IDA algorithms ---------- 34

 Preceding page blank

LIST OF ILLUSTRATIONS

Figure

1. Substack bases ---------- 23

2. Schematic diagram of the stacking process ---------- 24

3. Partitions of a module ---------- 29

4. Sample load in a lateral ---------- 30

5. Sample final partitioning of a module ---------- 31

6. Schematic diagram of the length-modular algorithm ---------- 33

Preceding page blank

I. INTRODUCTION

A. BACKGROUND

Analyses of transportation and storage problems have led to the development of many computer algorithms for the simulation of loading cargo into containers.[1] The usual objective of such simulations is to determine the number of containers required for a given list of cargo. This is vital information in the analysis of container dimensions, composition of transportation fleets, etc.; and computer simulation makes it relatively easy to perform the necessary parametric studies.

Loading simulations are particularly useful in military logistics problems. Accurate determination of the number of ship and aircraft sorties required for a given logistic operation allows a meaningful trade-off to be made between the time required for the operation and the number of transport vehicles to be assigned. Commercial problems also have this same trade-off situation, but simulation may not be as necessary because past experience with similar loading situations often provides the needed information.

Military planners frequently need sortie data for loading situations which exist only on paper. Proposed new transport vehicles and transported vehicles have increased the use of loading simulations in order that new logistic situations may be evaluated. For this reason, the majority of loading simulations have been conducted by military transportation agencies and their civilian contractors -

[1] The word "container", when used in this paper, refers to anything which holds goods, whether warehouse, parking lot, or the cargo carrying section of any vehicle.

 Preceding page blank

Boeing, Douglas, Lockheed, RAC, Rand, Mitre, and others - to solve military problems; but the methods used are also applicable to commercial problems.

The most sophisticated loading algorithms, such as those compared by Eastman and Holladay [1] attempt to fit the cargo into the containers in much the same fashion as loading personnel do. The simulation results compare closely with the results obtained by loadmasters in the field, particularly when the cargo consists of vehicles and large pallets.

B. THE NEED FOR BETTER LOADING ALGORITHMS

There are two shortcomings of these sophisticated algorithms which limit their applicability to some future loading problems. Nearly all of them consider only the two-dimensional problem of loading the container floor area. This was due primarily to their development for vehicle airlift problems. The sizes and weights of the transported vehicles and the low heights of the operational aircraft cargo compartments made stacking infeasible for the problems of major interest. Furthermore, the algorithms usually attempt to predict, rather than improve, the performance of loadmasters [1]. Most studies assume that the loadmaster's role in measuring and fitting cargo into vacant spaces is indispensable and that his methods leave little room for improvement. These assumptions become more questionable as the size of the airlifted force increases.

The advent of the jumbo aircraft, particularly the C-5, has placed new emphasis on loading algorithms. The proposed greater reliance on airlifting military forces in response to threats has

generated a need for improvement in the algorithms in order to make airlifting as effective as possible.

Two important goals of effective airlifting are rapid loading and efficient use of aircraft capacity. The present methods of actually loading aircraft cause these goals to conflict. A loadmaster usually obtains decreased container capacity utilization as the time allowed for loading decreases, ceteris paribus. There is an excellent chance that computers with efficient loading algorithms can achieve both goals much better than unaided human loadmasters, particularly when large items are to be loaded.

Decreased loading time could be achieved by using loading algorithms to provide computer printouts of instructions to loadmasters detailing exactly where each item is to be located in each aircraft. This would all but eliminate the time-consuming trial and error loading techniques presently employed. Even when late arrival of cargo or aircraft make the current instructions useless, a nearby computer terminal could be used to produce new instructions rapidly.

In addition to speeding the loading times, the loading algorithms should utilize container capacities as efficiently as practicable, preferably surpassing the present trial and error methods.

Some possible benefits of successfully developing computerized loading instructions include:

(1) The time to airlift a given military force would be reduced because of decreased loading time.

(2) The number of aircraft required for a given airlift capability would be reduced, lowering total procurement and operating costs.

(3) The congestion at the origin, enroute, and at destination airfields would be reduced because of fewer sorties required for a given airlift operation. This should further reduce the time and cost of an operation.

For effective competition with loadmasters, three-dimensional loading algorithms will have to receive much greater emphasis than they have in the past. Computerized instructions are needed for stacking and loading not only boxes and crates but also vehicles.

The greater cargo compartment height of the C-5 has aroused interest in stacking the smaller and lighter military vehicles, such as jeeps, trailers, mechanical mules, etc., by designing them with lower profiles and using racks and loading frames which may be loaded and unloaded easily. The space in an aircraft above most of the loaded vehicles is presently rarely used; and, as a consequence, the binding constraint on container capacity is usually loaded floor area, rather than volume or weight [1].

Attempts to produce computerized loading instructions could be beneficial even if they are not completely successful. It might be feasible to computerize only the loading of the larger items; even this would greatly simplify the loadmaster's task. Perhaps only the two-dimensional problems are appropriate for computerization, which would leave the stacking decisions to the loadmaster. The search for efficient algorithms might result in some new ground rules which would give loadmasters a better method of practicing their art.

C. RELIANCE ON HEURISTIC METHODS

The major problem in developing efficient loading algorithms is the non-applicability of existing mathematical programming techniques for efficiently utilizing two- and three-dimensional space. Knapsack problem solution techniques must be allowed to choose how many of each size of items are loaded, thus leaving some items on a given cargo list unloaded. Moreover, the best two-dimensional knapsack problem solution technique requires that the container floor be divided into rectangles of known dimensions before the technique can be applied [2].

Cutting stock problem solution techniques have some applications in container utilization efficiency. Techniques for optimal solutions to one-dimensional problems of fitting a given list of cargo into the minimum number of containers have been developed [3]. Their use is limited to those cases where assumptions can be made about how the second and third dimensions constrain loading; e.g., stacking is not possible, exactly two items can always be loaded side by side, etc. Two- and three-dimensional cutting stock solution techniques will select the best of many patterns submitted for consideration, but the formulation of patterns is an intractable problem for even a small number of different object sizes [4].

Most airlifts have enough different sizes of items to be loaded to easily violate some of the assumptions which must be made before present mathematical programming techniques can be used to minimize the number of aircraft sorties required. It is possible that new mathematical programming techniques will be developed to maximize utilization of two- and three-dimensional space with less

restrictive assumptions, but the likelihood of success in the near future seems to be very low.

The most promising method of developing efficient loading algorithms seems to be to study the effects of many collections of loading decision rules in order to ferret out those which lead to highest space utilization. Attempting to achieve the highest possible efficiency could become an endless task; a more reasonable goal would be to surpass the current efficiency of loadmasters.

The many alternatives an algorithm can take when another item is to be loaded could eventually be separated into three groups: those most likely to increase efficiency, those most likely to decrease efficiency, and those whose effects can not be safely predicted. This triage alone would be a major step forward in the quest for efficient algorithms.

D. A STANDARD FOR LOADING ALGORITHM COMPARISONS

Published algorithms with proven success in predicting the loadmaster's efficiency were sought for standards of comparison. Only one was found; it was part of a two-dimensional loading model developed by the Institute for Defense Analyses (IDA) to determine the number of aircraft sorties needed to haul a given list of cargo [1]. The algorithm, known as LOAD VEHICLES, loads rectangles (vehicles) into a larger rectangle (cargo compartment floor). It is frequently used for airlift simulations where the "highest area efficiency practicable" is desired. Several other loading models use variations of this algorithm. Boeing Aircraft's SLAM program, whose efficiency is compared with the IDA model in Ref. 1, is an example.

II. OBJECTIVES AND SCOPE

The objectives of this study were to develop a volumetrically efficient three-dimensional loading algorithm and an area efficient two-dimensional loading algorithm. Such algorithms are the first step in computerizing loading instructions. It was considered desirable, but not essential, that any algorithms developed would utilize container space as well as or better than the average loadmaster.

It was also considered desirable, but not necessary, that the algorithms developed be simple enough for use without a computer and that integer and linear programming be avoided if no significant efficiency would be lost thereby. This was accomplished, greatly reducing computer time, and it made the algorithms usable for hand calculations by loading personnel.

The three-dimensional problem was conceived as the filling of rectangular solid containers with rectangular solid objects so as to minimize the number of containers required to hold any given number of objects of many different sizes. Similarly, the two-dimensional problem involved only the rectangular floors of the containers and the rectangular bases of the objects to be loaded.

Container door dimensions were not considered. For simplicity, each piece of cargo was assumed to be marked "THIS END UP".

Cargo weight was not considered in the algorithms tested. The containers were assumed to be strong enough to support any weight placed anywhere. Center of gravity movement was disregarded. Each cargo item was assumed to be strong enough to support whatever other items that might be placed on top of it.

The algorithms, as presented, do not print out the locations of the cargo items; but simple modifications to the computer program on pages 40 through 43 allow this.

III. THE SYNTHETIC TEST LOADS

The desire to efficiently load all the items from a list containing different quantities of each of many different sizes of items required some method of selecting cargo dimensions and container dimensions. A group of dissimilar cargo lists was selected for testing proposed algorithms.

The unit of length measurement was set at six inches, and only integer values were used throughout the study. This was to restrict the number of different sets of dimensions to a more manageable group. A larger length unit would have reduced the group further, but it would also have reduced the accuracy with which the cargo items could be measured, since each item's dimensions are rounded up to the next integer unit.

A notional aircraft cargo compartment was selected with a length of 200 units, width of 30 units, and a height of 20 units. These dimensions were not changed throughout the study because the desired variety in the loads was achieved by varying the list of cargo.

Cargo lists were made by computer generation of one item at a time until a selected volume was exceeded. Two random selections were made from the integers 1, 2, . . . , 20, where each integer had an equal probability of being selected. The larger integer selected was the item's length; the smaller, the width. The height was similarly selected from 1, 2, . . . , 10. This produced 2,100 different sets of dimensions for items which could be on the cargo lists. The expected values of the length, width, and height were 13.825, 7.175, and 5.5 units, respectively.

Tables I and II show two distributions of generated cargo item volumes for four approximate total cargo volumes. Each table was compiled by generating enough items to exceed 80,000 cubic units of volume and the distribution of item volumes was recorded. Next, more items were generated and their volumes added to the 80,000 until 120,000 cubic units was exceeded. The recording and generation of additional items were continued using total volume increments of 40,000 cubic units until data was obtained for 200,000 cubic units. These tables will be used to illustrate the results obtained by the algorithms presented below.

Many other cargo lists were generated and loaded, but the algorithms produced no interesting changes in results for the same approximate total volumes of cargo. A few tests with larger average dimensions for the cargo items confirmed intuition that loading efficiencies decreased for the two algorithms presented below.

APPROXIMATE TOTAL VOLUME		80,000	120,000	160,000	200,000
GENERATED TOTAL VOLUME		80,629	120,172	160,623	200,487
TOTAL NUMBER OF ITEMS		133	206	285	361
NUMBER OF ITEMS IN EACH VOLUME RANGE					
VOLUME IN CUBIC UNITS	0 - 49	11	16	24	37
	50 - 99	13	20	28	31
	100 - 149	11	17	23	27
	150 - 199	8	15	22	29
	200 - 299	20	23	35	43
	300 - 399	8	15	24	30
	400 - 599	15	26	32	41
	600 - 799	12	17	25	36
	800 - 999	12	16	23	24
	1,000 - 1,499	8	15	25	34
	1,500 - 1,999	5	9	11	16
	2,000 - 4,000	10	12	13	13

Table I. Sample distribution of cargo items by volume

APPROXIMATE TOTAL VOLUME		80,000	120,000	160,000	200,000
GENERATED TOTAL VOLUME		81,589	120,103	160,138	200,005
TOTAL NUMBER OF ITEMS		162	220	292	374
NUMBER OF ITEMS IN EACH VOLUME RANGE					
VOLUME IN CUBIC UNITS	0 - 49	20	25	30	46
	50 - 99	16	24	39	47
	100 - 149	17	24	30	37
	150 - 199	18	22	26	28
	200 - 299	15	18	22	29
	300 - 399	12	14	21	29
	400 - 599	18	27	34	43
	600 - 799	16	20	23	32
	800 - 999	5	9	14	19
	1,000 - 1,499	12	16	23	28
	1,500 - 1,999	6	8	16	20
	2,000 - 4,000	7	13	14	16

Table II. Sample distribution of cargo items by volume.

IV. THE STACKING ALGORITHM

A. GENERAL

The stacking algorithm to be presented below was selected as the simplest method of achieving a good three-dimensional loading algorithm. It reduces the three-dimensional problem to a two-dimensional problem by loading the cargo onto notional pallets which must then be loaded into the containers by any floor area loading algorithm.

The base of each notional pallet is the largest cargo item on that particular pallet, rather than literally a metal or wooden platform under the cargo. Therefore, each pallet base takes the dimensions of the base of its largest cargo item.

The objective of the stacking algorithm is to maximize stacking efficiency where:

$$\text{(1)} \qquad \text{STACKING EFFICIENCY} = \frac{\text{TOTAL CARGO VOLUME}}{\text{TOTAL PALLET AREA} \times \text{STACKING HEIGHT}}$$

Stacking height is a constant equal to container height throughout this paper. Thus, the algorithm achieves its objective by minimizing the total area of the notional pallets it must use for a given volume of cargo to be loaded.

Volumetric loading efficiency is the stacking efficiency of the stacking algorithm multiplied by the floor area loading efficiency obtained by the two-dimensional algorithm which loads the notional pallets into the aircraft. The use of a stacking algorithm to palletize

the cargo before anything is placed on the container floor simplifies efforts to improve volumetric efficiency because such efforts can be divided between two paths which are considerably less complicated than trying to search for improvements in some three-dimensional algorithm.

B. METHOD

The first step in the algorithm is to order all of the cargo items by base size (area or perimeter). Items with the same base size are ordered by height. The stacking begins with the largest item being designated "Stack #1"; its length, width, height, and base area are recorded. The second largest item is then compared with the top of Stack #1. The item is stacked on #1 if it does not overhang any side of the top and if it does not cause the stack's height to exceed the stacking height; otherwise, it becomes "Stack #2".

Whenever an item is loaded onto a stack, its base dimensions become the new dimensions for the top of the stack, and its height is subtracted from the stack's ceiling clearance to obtain the new clearance. The unused area on the previous top of the stack is used for a substack. The base of the substack is the larger area rectangle ABGH or AJED of Figure 1. Substacking is performed using the same rules of fit as stacking. Items are substacked from the cargo list until either the substack reaches the ceiling or the entire cargo list has been scanned for items yet to be loaded which will fit onto the substack. Substacks are not numbered. The algorithm "forgets" them after every effort to fill them is completed.

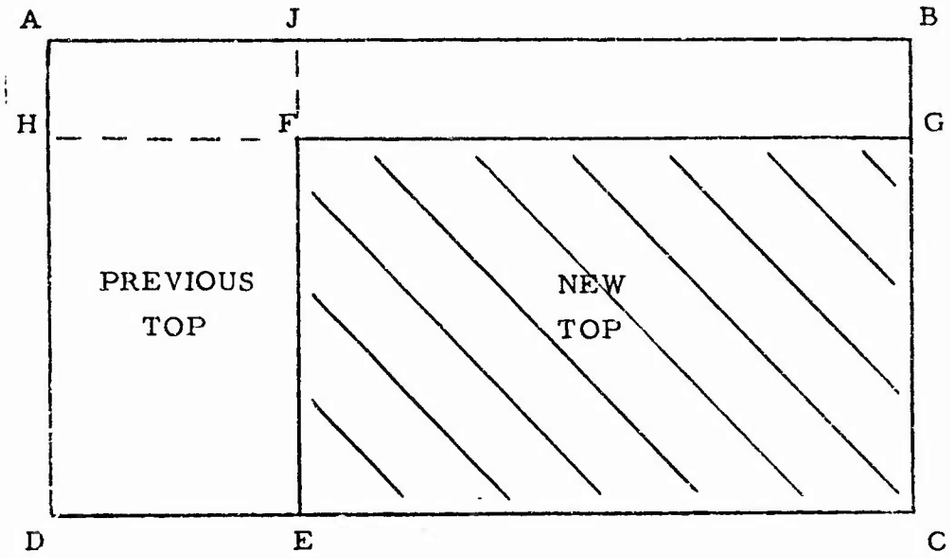

Figure 1. Substack bases

Stacking then resumes with the largest item remaining on the cargo list. An attempt is made to stack the item on the unfilled stacks in order of stack serial numbers, i.e., in the order in which the stacks were created. If the item fits one of the stacks, a substack is placed on that stack, if possible. If the item can not be placed on any unfilled stack, it becomes the base for a new stack.

The stacking algorithm continues until every item on the cargo list is positioned in a stack as the base, a member of the stack proper, or a member of one of the several substacks of the stack. Figure 2 is a schematic diagram of the stacking process.

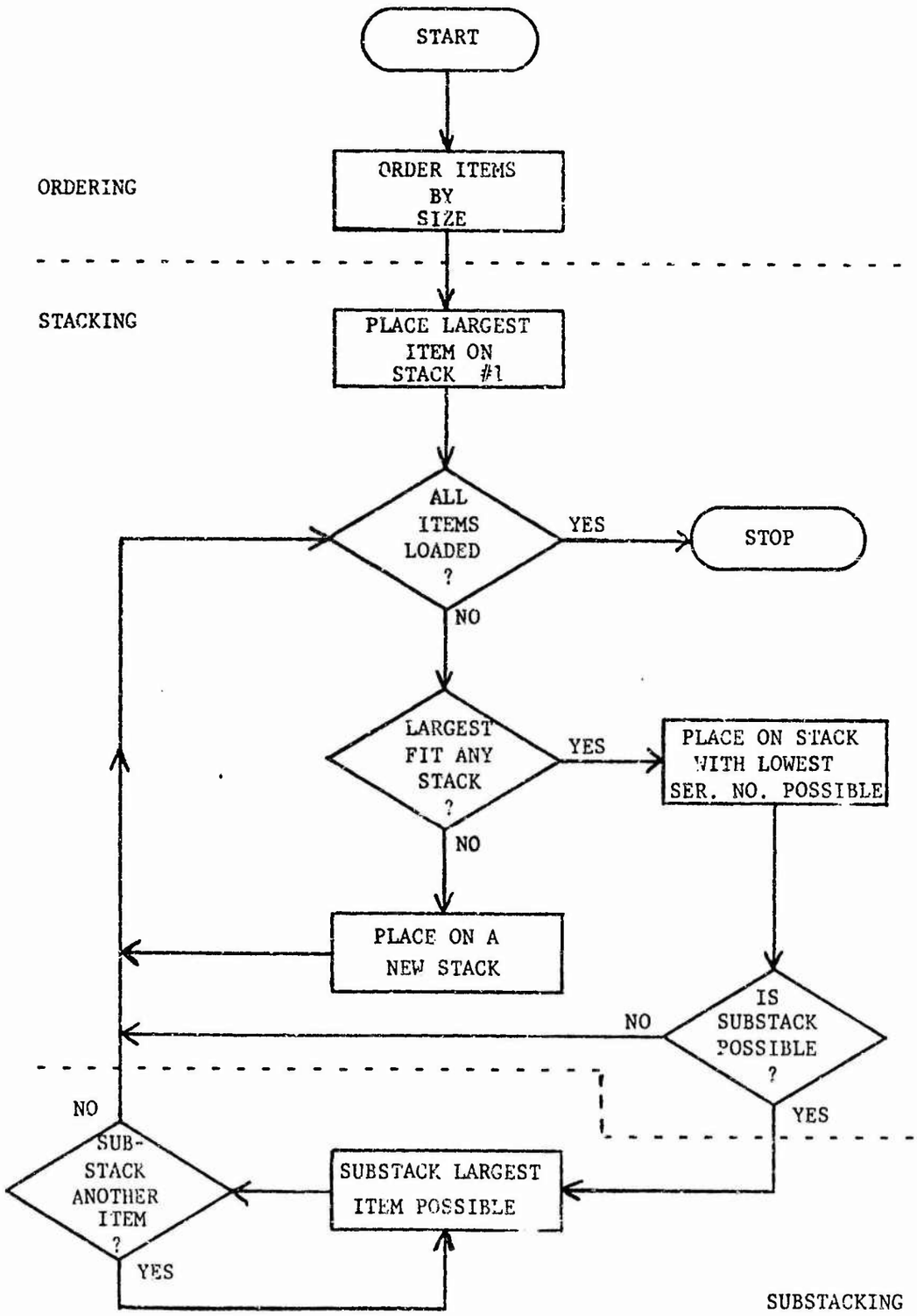

Figure 2. Schematic diagram of the stacking process

C. TEST RESULTS

The stacking algorithm proved to be quite efficient. Table III shows the stacking efficiencies obtained for the cargo volumes listed in Tables I and II. These results include both ordering the cargo items by base area and base perimeter.

APPROXIMATE CARGO VOLUME	80,000	120,000	160,000	200,000
CARGO FROM TABLE I				
STACKING EFFICIENCY (AREA ORDERING)	.908	.922	.938	.943
STACKING EFFICIENCY (PERIM. ORDERING)	.906	.927	.934	.941
CARGO FROM TABLE II				
STACKING EFFICIENCY (AREA ORDERING)	.913	.935	.945	.950
STACKING EFFICIENCY (PERIM. ORDERING)	.907	.918	.943	.950

Table III. Efficiencies of the stacking algorithm

The most interesting test result shown in Table III is the general tendency of stacking efficiency to increase with increased volume of cargo. This higher efficiency with higher volume was expected

because a larger number of items selected from a fixed number of different dimensions should cause the base area of each stacked item to more nearly cover the base area of the item below it in the stack.

Tests with cargo volumes between the tabulated volumes of Table III showed that this stacking efficiency increase is not monotonic as the table might imply. Additional cargo items were generated and added to those of Table I to produce a total cargo volume of 600,000 cubic units; this volume was stacked with 98.0% efficiency after area ordering.

This same increased efficiency phenomenon obviously made the use of substacking less important as cargo volume increased. Substacking increased the efficiency by about 2.5% for approximate total cargo volumes of 100,000 cubic units, but the increase was less than 1.0% when total cargo volume exceeded 400,000 cubic units. Substacking thus appears to provide very little extra efficiency for the synthetic test loads. It was not deleted from the algorithm because it can provide much larger efficiency increases in situations where there is a large difference in the base area of an item and the next smaller item on the cargo list.

Simulations were conducted with two realistic vehicle lists to determine how much use the stacking algorithm could make of the space in a C-5 if stacking were feasible. The first list was that for the 82nd Airborne Division, published in Ref. 1, containing 1,573 vehicles of 13 different types. The second list was for an infantry division and contained 6,811 vehicles of 132 different types. The first list was stacked with 80.0% efficiency and produced 763 stacks. When loaded by the IDA algorithm, these stacks required only 39

aircraft as opposed to 56 for the unstacked case. The second list resulted in 2,664 stacks with 84.0% efficiency. The IDA algorithm loaded these stacks into 186 aircraft vice 326 for the unstacked case.

At the end of all tests, the question of ordering the cargo items by base area or base perimeter was resolved in favor of base area. Area ordering usually, but not always, produced a slightly higher stacking efficiency than that obtained by perimeter ordering. Table III is typical of the results obtained for all cargo lists tested. Note that the one case where area ordering was not superior is in the 120,000 cubic units column.

D. SUMMARY

The stacking algorithm presented combines high stacking efficiency with computational ease. In the tests it provided increased efficiency as cargo volume was increased. It can be used with any two-dimensional loading algorithm to produce a three-dimensional algorithm.

A major obstacle to the use of stacking algorithms is the nature of the cargo. They will have limited applications for loading vehicles until engineering permits the stacking of some vehicles in airlift situations.

V. THE LENGTH-MODULAR ALGORITHM

A. GENERAL

The length-modular algorithm is so named because it divides each container into modules whose lengths are that of the longest cargo item in each module and whose widths are equal to container width. It is a two-dimensional loading algorithm which attempts to maximize area efficiency where:

$$(2)\quad \text{AREA EFFICIENCY} = \frac{\text{TOTAL PALLET AREA LOADED}}{\text{NUMBER OF CONTAINERS REQUIRED} \times \text{CONTAINER BASE AREA}}$$

As in equation (1), the pallet of equation (2) is notional. There may or may not literally be a wooden or metal platform under everything which covers some part of the container floor.

B. METHOD

The algorithm loads one container at a time. It first finds the length of the longest cargo item and uses that as the length of the first module. Whenever a module is created, it is partitioned into three rectangles as shown in Figure 3. Rectangle A is always completely covered with cargo, its length is always equal to that of the module, and its width is initially zero. Rectangle B, known as a submodule, is always empty and it initially covers the entire module. Rectangle C, known as a lateral, may be empty, partially filled, or completely filled; its initial dimensions are both zero. The lengths of rectangles A and B are measured in the same direction as the module length, but the length of rectangle C is always its longer dimension.

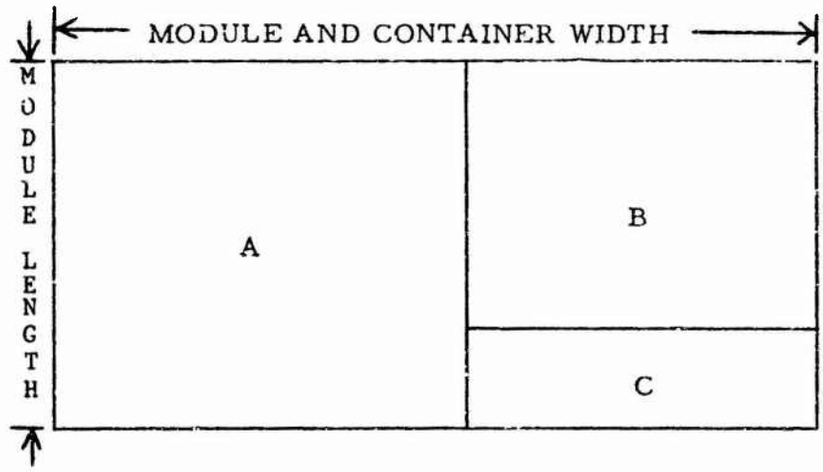

Figure 3. Partitions of a module

The first step in filling the module is to load the widest item with the same length as the module. The widths of rectangles A and B are respectively increased and decreased by the item's width. The algorithm continues loading the widest items of that length until the width of rectangle B is zero, all items of that length are loaded, or all items yet to be loaded of that length are wider than the width of rectangle B. Whenever the first case occurs, rectangle A occupies the entire module, the algorithm is finished with that module, and a new module is started.

If the first case does not occur, the algorithm finds the longest possible item which will fit into rectangle B. If none will fit, the algorithm is finished with that module and starts a new one. If some item does fit into rectangle B, say vehicle X, the rectangle's length is decreased to the length of vehicle X. The area lost by rectangle B because of this decrease becomes the area of rectangle C. Note that vehicle X has not been loaded yet. Rectangle C, the lateral, is then

loaded with the longest item possible, then the next longest possible, etc., until no items on the cargo list will fit lengthwise into the unloaded length of the rectangle. Rectangle C is loaded before rectangle B in order that the same computer routine which loads an empty module can load an empty submodule without having to return to rectangle C or remember its dimensions.

Figure 4 shows a sample filled lateral. Note that item 3 could have been rotated 90 degrees to make the unused length of the lateral longer. This rotation was found to be unproductive, as a rule, because laterals are almost always extremely narrow, on the order of two or three units wide for the synthetic loads. They would also normally be too narrow to hold the shortest vehicle in an actual load.

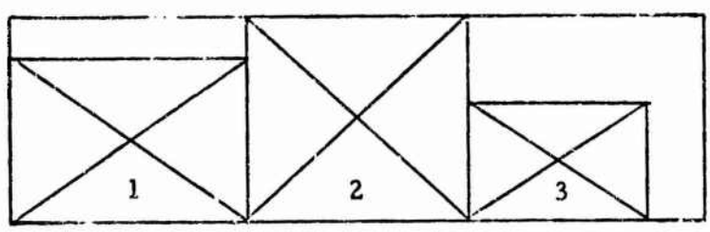

Figure 4. Sample load in a lateral

The algorithm then prepares to load rectangle B with the longest item possible. If this item is not vehicle X, which might have been loaded into rectangle C, or one of the same length, the length of rectangle B is further decreased to the length of this new longest item. The area lost in rectangle B is again added to rectangle C, but no effort is made to fill this area because this situation rarely occurs during a load of many items.

Rectangle B, the submodule, is then loaded in the same manner as the original module, i.e., it is partitioned into three rectangles, A', B', C', which are loaded in the same manner as the original A, B, C. This partitioning of submodules continues until some submodule of a submodule is too small for any unloaded item on the cargo list. The maximum number of partitionings of a module and its submodules is the module's length in integer units; the minimum is one, regardless of length. Four or more partitionings were extremely rare for the synthetic test loads.

Figure 5 illustrates how a module might look when the algorithm is finished with it. The rectangles A, A', A'', A''' are completely filled. The rectangles C, C', C'' may each be filled, partially filled, or empty. Rectangle B''' is empty, but its area could be zero.

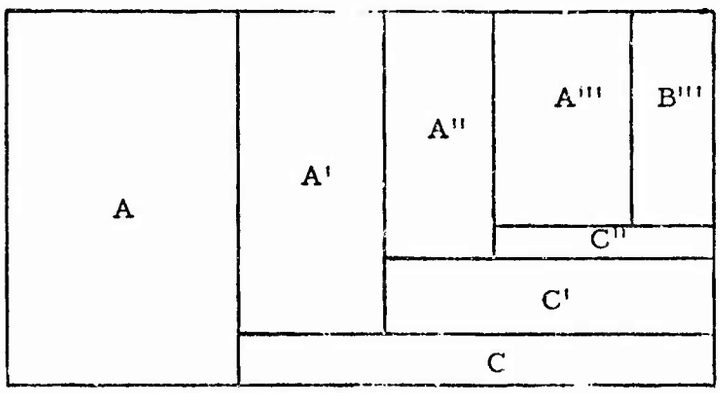

Figure 5. Sample final partitioning of a module

The second and succeeding modules in a container are given a length equal to the longest cargo item remaining on the list which is not longer than the remaining length of the container. This makes

each module no longer than any created before it in that container. The process of creating and loading modules continues until some module is created which is too short for all of the remaining items on the cargo list. If all items are loaded, the algorithm is finished; otherwise, it starts with a new container. Figure 6 is a schematic diagram of the length-modular algorithm.

C. TEST RESULTS

The length-modular algorithm was tested in conjunction with the stacking algorithm for many synthetic test loads. The stacking algorithm was usually applied first, and then the length-modular algorithm loaded the stacks into the containers.

Table IV shows the results of applying the length-modular algorithm to the pallets which were stacked and then tabulated in Table III. The results of loading the same pallets with the IDA algorithm are also presented in Table IV for comparison. Area efficiency was computed in each case by treating the last loaded container's length equal only to the loaded length of the container. This permitted a more meaningful comparison of the efficiencies to be made, since no more than three containers were ever required for the volumes in the preceding tables.

Area efficiency for the length-modular algorithm was found to increase with increased cargo volume in a manner similar to that of the stacking algorithm. A larger number of items selected from the 210 different sets of base dimensions usually enabled the algorithm to find a better fitting cargo item for a particular vacant space than when the number of items to be loaded was fewer.

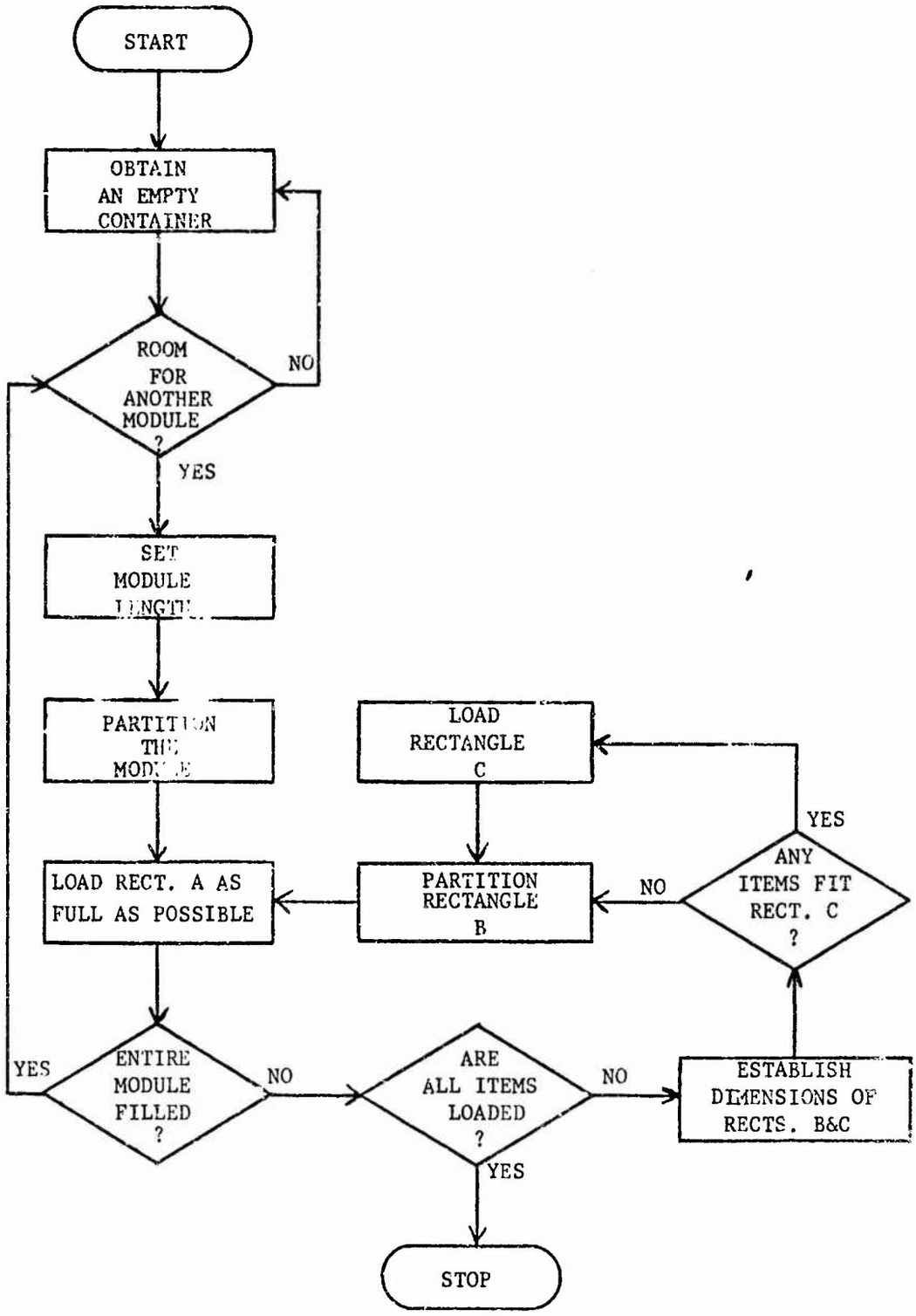

Figure 6. Schematic diagram of the length-modular algorithm

APPROXIMATE TOTAL CARGO VOLUME	80,000	120,000	160,000	200,000
TABLE I CARGO				
STACKING EFFICIENCY (AREA ORDERING)	.908	.922	.938	.943
AREA EFFICIENCY WHEN THESE STACKS WERE LOADED BY THE L-M ALGORITHM	.912	.935	.952	.965
AREA EFFICIENCY WHEN SAME STACKS WERE LOADED BY THE IDA ALGORITHM	.790	.780	.766	.767
TABLE II CARGO				
STACKING EFFICIENCY (AREA ORDERING	.913	.935	.945	.950
AREA EFFICIENCY WHEN THESE STACKS WERE LOADED BY THE L-M ALGORITHM	.903	.952	.943	.960
AREA EFFICIENCY WHEN SAME STACKS WERE LOADED BY THE IDA ALGORITHM	.767	.775	.781	.774

Table IV: Area efficiency comparison of the length-modular and IDA algorithms

Three other permutations of modular loading were tested. The first might be called "width-modular" because module length was set equal to the width instead of the length of the largest remaining item. The other two permutations were length-modular and width-modular with module width set equal to the container's length instead of width. These permutations gave much poorer results than the original method.

The same two vehicle lists discussed in Section IV were loaded unstacked by the length-modular algorithm, and comparisons were made with the IDA algorithm. The smaller list was loaded into 54 C-5's vice 56 for the IDA model. The larger list required only 310 aircraft vice the IDA model's 326.

It was interesting to note that the IDA model gave area efficiencies within $\pm$ 3.2% of 78.0% throughout about 40 synthetic loads of ten different volumes. However, when it loaded the two lists of vehicles, area efficiency increased to about 84% for both lists. Its performance did not ever equal that of the length-modular algorithm for any of the tests conducted, but there could be cases where it would be superior.

A few tests were conducted where the length-modular algorithm loaded the container floor first, and then the stacking algirithm loaded vertically upon those items which covered the floor. The area efficiency thus obtained was very high, never less than 96.7%, but the stacking efficiency was so degraded that volumetric efficiency was always 15-20% lower than when the same load was stacked before the container floor was covered.

D. SUMMARY

The length-modular algorithm has been shown to provide excellent area efficiency for some loading situations. It is simple in method, although the computer program on pages 40 through 43 is somewhat complicated by steps to simplify record keeping and reduce computer time. There are several places where the algorithm as listed sacrifices area efficiency in order to save time. It generally provides increased efficiency with increased total volume of cargo.

It should be noted that the modules in any loaded container can be repositioned to move the container's center of gravity longitudinally, and items within each module can be moved in several ways to move the center of gravity laterally. This feature of modular loading facilitates the algorithm's proposed use for computerizing loading plans for aircraft.

VI. AREAS FOR FURTHER STUDY

The particular rules and methods used in this study are only a minute part of what could be considered. The high efficiencies of the algorithms presented will not permit major increases, but some worthwhile increases in efficiency might be easily discovered in both algorithms. The areas suggested below for further study are only a few which might be promising for increased efficiency and inclusion of aspects such as weight and center of gravity.

A. METHODS WITH "THIS END UP" ASSUMPTION REMOVED

Since many cargo items may be loaded with any of its three axes vertical, it would be useful to know which axis should be placed vertically when the stacking algorithm is given a choice. Simple rules, such as prescribing the longest, shortest, or middle length axis, might be found to yield the highest volumetric efficiency. More complex rules, which select a different axis for different stack clearances or other stacking parameters, might be necessary.

B. PRE-STACKING

There are countless ways in which many items of one or more common dimensions might be combined into a rectangular solid having little or no wasted space. Such a solid would then be stacked as one item. This type of cargo list consolidation before the stacking algorithm is applied might have surprising advantages in efficiency and speed.

C. HEIGHT-MODULAR STACKING

It would be interesting to know what might be done by having a two-dimensional loading algorithm load modules with only items of particular height ranges and then stack the modules in various ways. Such modules would not necessarily cover the container floor.

D. WEIGHT CONSIDERATIONS

Modifications to the two algorithms presented could allow them to consider center of gravity constraints and total weight applied to any part of the container floor. This will be necessary before computerized loading instructions can become a reality for aircraft. The modifications might not significantly decrease the efficiencies of the algorithms.

E. NON-RECTANGULAR CONTAINERS

Methods for loading non-rectangular containers with rectangular cargo have received even less attention than the rectangular container case. Aerodynamic, hydrodynamic, and other engineering considerations dictate that many containers take on shapes which will <u>always</u> result in some wasted space for any realistic non-fluid load. Minimization of the wasted space would be a real challenge.

VII. CONCLUSIONS AND RECOMMENDATIONS

The two algorithms presented have demonstrated high volumetric and area efficiencies for loading a large number of items of many different sizes. They are an important first step in computerizing the loading of aircraft and other containers. The length-modular algorithm is the more important of the two for major airlifting problems because they presently involve cargo which permits little stacking.

Much remains to be done before computerized loading can become a reality. Loading algorithms must be able to consider each item's weight and fragility, as well as its effect upon the container's center of gravity. Methods for utilizing container space more efficiently should be sought, but the algorithms presented here should be good enough for come of the interim work necessary for development of computerized loading systems.

At this stage, a logical next step would be to see how well the length-modular algorithm can compete with loadmasters in two-dimensional loading of vehicles without weight constraints. If the IDA loading algorithm is truly an accurate predictor of human loading efficiency, then chances are excellent that the length-modular algorithm can make computerized aircraft loading a reality.

COMPUTER PROGRAM

```
C    THE NPS LOAD MODEL

C    DETERMINES THE NUMBER OF CONTAINERS TO HOLD THE INPUT
C    LIST OF ITEMS

C    LIMITATIONS: 7,000 ITEMS OF LENGTH NOT GREATER THAN 100
C                 UNITS, WIDTH OF 40 UNITS, HEIGHT OF 99 UNITS,
C                 UP TO 300 DIFFERENT TYPES OF ITEMS ALLOWED

C                   UP TO 999 CONTAINERS ALLOWED, ALL MUST HAVE
C                   SAME DIMENSIONS, LENGTH NOT GREATER
C                   THAN 999 UNITS, WIDTH 99, HEIGHT 99

C    FIRST DATA CARD HAS NUMBER OF ITEM TYPES AND CONTAINER
C    LENGTH, WIDTH, AND HEIGHT IN FOUR TEN COLUMN FIELDS
C    ALL OTHER DATA CARDS HAVE NUMBER OF ITEMS OF THAT TYPE
C    AND THE LENGTH, WIDTH, AND HEIGHT OF THAT TYPE IN FOUR
C    TEN COLUMN FIELDS



C   INITIALIZING

      DIMENSION IL(300), IW(300), IH(300), ML(1600),
     1MW(1600), MH(1600), N(300), LL(1600), LW(1600),
     2KL(1600), KW(1600), IS(300), NE(1600), LH(1600),
     3MTX(100,41)
      DO 600 K = 1, 100
      DO 600 L = 1, 41
 600  MTX(K,L) = 0
      NV = 0
      JP = 0
      IB = 0
      READ(5,91) NTI, IAL, IAW, IAH
  91  FORMAT(4I10)
      DO 954 K = 1, NTI
      READ  (5,91) M, IRL, IRW, IRH
      NV = NV + M * IRL * IRW * IRH
      IB = IB + M
      JP = JP + 1
      IL(JP) = IRL
      IW(JP) = IRW
      IH(JP) = IRH
      N(JP) = M
 954  IS(JP) = JP

C   COMPLEXITY SORTING
      JD = JP - 1
      DO 100 J = 1, JD
      M = IS(J)
      IQ = J
      JW = J + 1
      DO 200 I = JW, JP
      MG = IS(I)
      IF ( IL(M) * IW(M) - IL(MG) * IW(MG) ) 40, 50, 200
```

```
   50  IF ( IH(M) .GE. IH(MG) ) GO TO 200
   40  M = MG
       IQ = I
  200  CONTINUE
       IDUM = N(IQ)
       N(IQ) = N(J)
       N(J) = IDUM
       IS(IQ) = IS(J)
       IS(J) = M
  100  CONTINUE

C  COMMENCE STACKING
       DO 10 K = 1, IB
       ML(K) = 100
       MW(K) = 40
       LH(K) = IAH
   10  MH(K) = IAH
       LA = 0
       M = 0
       L = 1
       DO 400 MG = 1, JP
       IF ( N(MG) .EQ. 0 ) GO TO 400
       IQ = N(MG)
       DO 400 J = 1, IQ
       JE = M + 1
       I = IS(MG)
       DO 300 JZ = L, JE
       K = JZ
       IF ( IL(I) .LE. ML(K) .AND. IW(I) .LE. MW(K) .AND.
      3IH(I) .LE. MH(K) ) GO TO 56
  300  CONTINUE
   56  IF ( MH(K) .EQ. IAH) GO TO 60
       LH(K) = MH(K)
       IF ( ( MW(K) * ( ML(K) - IL(I) ) ) .GE. ( ML(K) *
      4( MW(K) - IW(I) ) ) ) GO TO 116
       LL(K) = ML(K)
       LW(K) = MW(K) - IW(I)
       GO TO 121
  116  LL(K) = MW(K)
       LW(K) = ML(K) - IL(I)
       IF ( LL(K) .GE. LW(K) ) GO TO 121
       IDUM = LL(K)
       LL(K) = LW(K)
       LW(K) = IDUM
  121  DO 440 IC = MG, JP
  128  IF ( N(IC) .EQ. 0 ) GO TO 440
       IF ( LH(K) .EQ. 0 ) GO TO 60
       IE = IS(IC)
       IF ( IL(IE) .GT. LL(K) .OR. IW(IE) .GT. LW(K) .OR.
      5IH(IE) .GT. LH(K) ) GO TO 440
       N(IC) = N(IC) - 1
       LL(K) = IL(IE)
       LW(K) = IW(IE)
       LH(K) = LH(K) - IH(IE)
       IF ( LH(K) .GE. IH(IE) ) GO TO 128
  440  CONTINUE
   60  ML(K) = IL(I)
       MW(K) = IW(I)
       MH(K) = MH(K) - IH(I)
       N(MG) = N(MG) - 1
       IF ( MH(L) .EQ. 0 ) L = L + 1
       IF ( M .GE. K ) GO TO 400
       M = M + 1
       LA = LA + ML(K) * MW(K)
       KL(K) = ML(K)
       KW(K) = MW(K)
       IC = KL(K)
       ID = KW(K)
       MTX(IC,41) = MTX(IC,41) + 1
       MTX(IC,ID) = MTX(IC,ID) + 1
  400  CONTINUE
```

```
      WRITE (6,80) IB, NV, M, L, LA
   80 FORMAT ( 1H1,  'NUMBER OF ITEMS LOADED = ', I4//
     7 ' VOLUME LOADED = ', I8//
     8  ' NUMBER OF STACKS = ', I4//,
     9  ' SERIAL NUMBER OF FIRST INCOMPLETE STACK = ', I4//
     1  ' FLOOR SPACE COVERED = ', I7// )
      WRITE (6, 82) KL(1), KW(1), KL(M), KW(M)
   82 FORMAT ( ' FIRST STACK BASE LENGTH = ', I3, 4X,
     2 ' WIDTH = ', I2, 4X,// ' LAST STACK BASE LENGTH = ',
     3 I3, 4X, ' WIDTH = ', I2// )
      WRITE (6,90) ML(1), MW(1), MH(1), ML(M), MW(M), MH(M)
   90 FORMAT ( ' TOP OF FIRST STACK LENGTH = ', I3, 4X,
     3 ' WIDTH = ', I2, 4X, ' CLEARANCE = ', I2,//
     4' TOP OF LAST STACK LENGTH  = ',
     4  I3, 4X, ' WIDTH = ', I2, 4X, ' CLEARANCE = ', I2// )

C   COMMENCE LOADING
      NAC = 0
      ISL = 100
  640 IF ( MTX(ISL,41) .GT. 0 ) GO TO 650
      IF ( ISL .EQ. 1 ) GO TO 99
      ISL = ISL - 1
      GO TO 640
  650 LPA = 0
      NAC = NAC + 1
      NL = IAL
      KSL = ISL
  810 NW = IAW
      IF ( NL .EQ. 0 ) GO TO 800
      IF ( NL .LT. KSL ) KSL = NL
  630 IF ( MTX(KSL,41) .GT. 0 ) GO TO 840
      IF ( KSL .EQ. 1 ) GO TO 800
      KSL = KSL - 1
      GO TO 630
  840 NL = NL - KSL
      KSW = KSL
  710 IF ( KSW .GT. NW ) KSW = NW
  660 IF ( MTX(KSL,KSW).GT. 0 ) GO TO 820
      IF ( KSW .EQ. 1 ) GO TO 780
      KSW = KSW - 1
      GO TO 660
  820 LPA = LPA + KSW * KSL
      MTX(KSL,KSW) = MTX(KSL,KSW) - 1
      MTX(KSL, 41) = MTX(KSL, 41) - 1
      NW = NW - KSW
      IF ( NW .NE. 0 ) GO TO 710
      GO TO 810
  780 IF ( KSL .EQ. 1 ) GO TO 800
      LSL = KSL - 1
  740 JW = 1
      JL = NW
  700 IF ( MTX(LSL,41) .GT. 0 ) GO TO 667
  634 IF ( LSL .EQ. 1 ) GO TO 810
      LSL = LSL - 1
      JW = JW + 1
      GO TO 700
  667 DO 633 J = 1, NW
      IF ( MTX(LSL,J) .GT. 0 ) GO TO 670
  633 CONTINUE
      GO TO 634
  670 IF ( JL .GE. JW ) GO TO 730
      IDUM = JL
      JL = JW
      JW = IDUM
  730 JSL = ISL
  760 IF ( JSL .GT. JL ) JSL = JL
  680 IF ( MTX(JSL,41) .GT. 0 ) GO TO 720
  770 IF ( JSL .EQ. 1 ) GO TO 750
      JSL = JSL - 1
      GO TO 680
  750 LSW = LSL
```

```
790   IF ( LSW .GT. NW ) LSW = NW
850   IF ( MTX(LSL,LSW).GT. 0 ) GO TO 860
      IF ( LSW .EQ. 1 ) GO TO 870
      LSW = LSW - 1
      GO TO 850
870   IF ( LSL .EQ. 1 ) GO TO 810
      LSL = LSL - 1
      GO TO 740
720   JSW = JSL
      IF ( JSW .GT. JW ) JSW = JW
880   IF ( MTX(JSL,JSW).GT. 0 ) GO TO 890
      IF ( JSW .EQ. 1 ) GO TO 770
      JSW = JSW - 1
      GO TO 880
890   JL = JL - JSL
      MTX(JSL, 41) = MTX(JSL, 41) - 1
      MTX(JSL,JSW) = MTX(JSL,JSW) - 1
      LPA = LPA + JSW * JSL
      IF ( JL .EQ. 0 ) GO TO 750
      GO TO 760
860   NW = NW - LSW
      MTX(LSL,LSW) = MTX(LSL,LSW) - 1
      MTX(LSL, 41) = MTX(LSL, 41) - 1
      LPA = LPA + LSW * LSL
      IF ( NW .EQ. 0 ) GO TO 810
      GO TO 790
800   PLA = LPA
      AEF = PLA / ( IAL * IAW )
      WRITE (6,900) NAC, LPA, NL, AEF
900   FORMAT(' A/C NO. ', I3, 5X, ' AREA LOADED = ', I5, 5X,
     6 ' UNUSED LENGTH = ', I3, 5X, ' AREA EFFICIENCY = ',
     7F4.2///)
      GO TO 640
 99   AL = LA
      TVS = NV / ( AL * IAH )
      TAS = AL / ( NAC * IAL * IAW )
      TTV = TVS * TAS
      WRITE (6, 57) TVS, TAS, TTV
 57   FORMAT ( ' TOTAL VERTICAL EFFICIENCY   = ', F4.2,//
     1 ' TOTAL AREA EFFICIENCY        = ', F4.2//
     2 ' TOTAL VOLUMETRIC EFFICIENCY = ', F4.2,///)
      STOP
      END
```

LIST OF REFERENCES

1. Eastman, S. E. and Holladay, J. C., "Aircraft Loading Considerations: A Sortie Generator for Use in Planning Military Transport Operations," Naval Research Logistics Quarterly, Volume 15, pages 99-119, (1968).

2. Gilmore, P. C. and Gomory, R. E., "The Theory and Computation of Knapsack Functions," Operations Research, Volume 14, pages 1045-1074, (1966).

3. Gilmore, P. C. and Gomory, R. E., "A Linear Programming Approach to the Cutting Stock Problem. Part II," Operations Research, Volume 11, pages 863-888, (1963).

4. Gilmore, P. C. and Gomory, R. E., "Multi-Stage Cutting Stock Problems of Two and More Dimensions," Operations Research, Volume 13, pages 94-120, (1965).

UNCLASSIFIED
Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) Naval Postgraduate School Monterey, California 93940	2a. REPORT SECURITY CLASSIFICATION Unclassified 2b. GROUP

3. REPORT TITLE

Area-Efficient and Volume-Efficient Algorithms for Loading Cargo

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)
Master's Thesis; September 1970

5. AUTHOR(S) (First name, middle initial, last name)

Ernest Larry DeSha

6. REPORT DATE September 1970	7a. TOTAL NO. OF PAGES 45	7b. NO. OF REFS 4
8a. CONTRACT OR GRANT NO. b. PROJECT NO. c. d.	9a. ORIGINATOR'S REPORT NUMBER(S) 9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)	

10. DISTRIBUTION STATEMENT

This document has been approved for public release and sale; its distribution is unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY Naval Postgraduate School Monterey, California 93940

13. ABSTRACT

The proposed greater reliance upon airlifting military forces demands that cargo loading time be minimized while utilization of aircraft cargo compartment space is maximized. Two loading algorithms have been developed with these goals in mind - a two dimensional one for loading cargo where all items must be placed on the floor, and a three-dimensional one for cargo which can be stacked. The three-dimensional algorithm consists of the two-dimensional algorithm and a special stacking algorithm. Tests using randomly generated three-dimensional cargo lists indicate that 90% area efficiencies for the two-dimensional and 80% volumetric efficiencies for the three-dimensional algorithm are possible. These algorithms were designed for either hand calculations or computer calculations.

DD FORM 1 NOV 65 1473 (PAGE 1)
S/N 0101-807-6811

UNCLASSIFIED
Security Classification
A-31408

UNCLASSIFIED

Security Classification

14. KEY WORDS	LINK A ROLE	LINK A WT	LINK B ROLE	LINK B WT	LINK C ROLE	LINK C WT
Loading						
Packing						
Containerization						
Airlift						

DD FORM 1 NOV 65 1473 (BACK)

S/N 0101-807-6821

UNCLASSIFIED

Security Classification

A-31409