International Journal of Trend in Research and Development, Volume 3(1), ISSN: 2394-9333
www.ijtrd.com

Evolutionary Approach for Optimizing 3D Heterogeneous Bin Packing Problems

[1]A.D. Jaisree, [2]R. UmaGowri and [3]S.K. Rajesh Kanna
[1]PG Student, [2]Assistant Professor, [3]Professor.
[1,2]Department of Computer Science and Engineering, [3]Department of Mechanical Engineering,
[1,2]Mahindra Engineering College, Salem, India
[3]Rajalakshmi Institute of Technology, Chennai, India

Abstract: This paper presents a combinational Hybrid Genetic Algorithm (HGA) with packing tuning approach for solving Three Dimensional (3D) Single container arbitrary sized heterogeneous bin packing optimization problem, by considering practical constraints in the shipment container loading industries. Aim of this paper is to (i) pack 3D arbitrary sized heterogeneous bins in to a container. (ii) Improve packing by optimizing the empty volume inside the container using genetic approach. (iii) obtain a feasible packing pattern, various practical constraints like box orientation, stack priority, container stability, weight constraint, overlapping constraint, and shipment placement constraint were also considered. (iv) Tuning algorithm was used for sequential packing without gap. 3D container loading problem consists of 'n' number of boxes to be packed in to a container of standard dimension in such a way as to maximize the volume utilization and in turn profit. Furthermore, Boxes to be packed are of various sizes and of heterogeneous shapes. In this research work, several heuristic GA operators were proposed to solve the container loading problem that significantly improve search efficiency and help to load most of the heterogeneous boxes into a container along with the optimal position of loaded boxes, and aid box orientation with less computational time. Tuning algorithm was used to make the genetic output in to packing pattern in an understandable format and without empty space in less computational time. In general, combination of Hybrid GA in conjunction with the tuning algorithm is substantially better and more satisfactory than those obtained by applying heuristics to the bin packing directly.

Keywords: *Hybrid Genetic Algorithm; Container Loading problem; Tuning Algorithm; Optimization*

I. INTRODUCTION

Due to industrial revolution, today, most of the industries are focusing towards globalization. Globalized market results in rapid development of international trade and in turn corporations have substantially increases the scope and magnitude of their global production and distribution networks. So a need arises for those firms to deliver and distribute their goods to customers, warehouses, and to distribution centers all around the world by means of cargo transport. For cargo transport, type and number of containers rented from shipping or from air cargo is a major issue which influences the cost of that product and profit margin of the firm. Because each container has its own volume and weight limits, based on which freight rates will be calculated by cargo transporters, that freight rate is also included in the product cost. In general, the freight rate for each container is sum of fixed cost for using that container and variable cost that depends on the total weight of items to be packed. Thus, it becomes clear that the product cost increases with an increase in the cargo cost.

During packing of bins, some empty space may be wasted inside the container because all items need not be of same shape and size. This empty space leads to instability, usage of additional container, airbags, etc. In turn, there will be an increase in the freight cost, transport cost, revenues for exporters, product cost, etc. So in this research work, container loading problem was taken up and solved.

In the light of the above, it is necessary to pack boxes tightly inside the container with less wastage of free space. Traditionally, packing problems were resolved through intuition, experience and judgments [100] manually, but this decision making approach lacks systemization and it was not known how effective these decisions were. Basically, no fast algorithm allows an exact solution for perfect packing. So this research work is focused towards a heuristic genetic approach which is the best among evolutionary algorithms to solve complex issues in optimization with difficult constraints.

In rest of this paper, literature review on Genetic Algorithms and Bin Packing are stated in Section 2. Then, methods of BP are discussed in Section 3 and 4. In section 5, GA concept and implementations are explained. In Section 6, system implementation of GANP will be shown. Implemented Practical constraints are discussed in section 7. Finally, results and conclusion of the entire study are presented.

II. LITERATURE SURVEY

In the year 1965, Golmore proposed stack building approach for cutting problem optimization. Historically, the first container loading approach was proposed by Christofides, Mingozzi, and Toth in 1979 for solving the liquid loading problem [3]. This algorithm used dominance criterion and assumed that those liquids will not mix with each other. It should be noted that due to the vast number of publications and researches addressing container loading problem, this review is necessarily restricted to the key relevant ideas of genetic algorithm for container loading. A procedure for packing boxes into a container using layer-by-layer filling concept was first developed by Gorge and Robinson in 1980. After that, most of the researches were focused towards the development of algorithms using wall and layer building concepts for maximizing container volume

utilization [6], but not concentrating on practical constraints. Only a limited number of researches have dealt with container loading problems along with practical constraints with some specific assumptions. Gehring et al in 1990 developed heuristics by considering the simple practical constraints for container loading [7]. However, in recent years, researchers found that only the hybrid algorithm along with intelligent algorithms gives the optimal packing of boxes by satisfying practical constraints also. Thus it initiates a basic idea and opens a new era for successive research on intelligent algorithms. Recent research diverted towards Meta heuristic algorithms like Tabu search, Genetic Algorithm, simulated annealing, etc to be applied to the container loading problem [3]. Bortfeldt and Gehring in 2001 proposed a hybrid GA based on the layer concept for solving container loading problem with several practical constraints [11]. In addition to these evolutionary heuristics, a few papers proposed an analytical approach, attempting to find a near optimal solution. Interestingly a majority of these evolutionary approaches are hybrid / heuristic, i.e. combining the GA with other algorithms or by modifying the GA. A review of relevant literatures confirms that the evolutionary approach works very well when compared to traditional optimization procedures and the computational time was also reduced significantly [7]. Various algorithms given in the literature for bin packing are the First Fit Decreasing Algorithm, Best-Fit Decreasing Algorithm [12], Improved First Fit Algorithm, Wall Building Algorithm [6], Layer-by-Layer Algorithm, Matrix Methods, Tuning Algorithm, Greedy Algorithm, Tree Search Algorithm, Traditional Optimization Algorithms, Stack-Building Algorithm, Guillotine Cutting Algorithms, Cuboids Arrangement Algorithms, etc.
As the result of this literature survey, in this research work a 3D single container with 'n' number of arbitrary sized heterogeneous box loading using a Hybrid Genetic Algorithm that would satisfy many practical constraints was implemented.

III. BIN PACKING PROBLEM

A Container is a large rectangular box of standard dimension used to pack boxes and to transport them from one location to another. Identifying the best mix of boxes inside the container to utilize its maximum volume is generally called as the optimal container loading problem or bin packing problem [3]. In general, the container loading problem is normally classified based on the number of containers and type of boxes to be loaded, as single/multiple container loading and homogenous/heterogeneous problem respectively [9]. Bin-packing has been defined in several different forms depending on the application [16]. Bin-packing has many aliases, such as stock cutting, vehicle loading, air container loading, scheduling, and knapsack. Many of the 3D bin-packing problems have been reduced to two dimensional or even one dimensional to solve, but these types of problems were not acceptable because of non-reasonable representations, specific assumptions and constraints especially for the GABP [3].

In this research work, the problem related to a 3D single container loading with weakly heterogeneous problem was taken and solved. Bin shapes considered are rectangular, cubical, cylindrical, and spherical with varying dimensions, which will be discussed later in section 7. The objective of the container loading problem is to pack the given 'n' number of boxes, each weight Wi and volume Vi, into the container of capacity 'C' without exceeding it and also without violating practical constraints like box orientation, weight, stability, placement, and overlapping.

IV. HEURISTIC APPROACHES

To solve this type of problems, simple heuristic and hybrid algorithms, used by most of the researchers concentrate on volume maximization and in turn cost/profit [101, 102], but lack in considering the technological and practical constraints. So the results obtained by these algorithms were good but unfortunately, their effectiveness was very limited and they were not feasible for bin packing, to which pervious researchers had not paid much attention. In recent years, researchers focused towards solving practical constraints and found that evolutionary algorithms will solve above mentioned problem [6]. Again most of the evolutionary algorithms are time consuming while considering practical constraints. In today's globalized competitive market, time places a vital role and thus, a necessity arises to reduce the computational time to find the optimal solution, researchers therefore focused towards the Genetic Algorithm which becomes a popular choice for numerous difficult optimization problems where conventional methods tend to fail [1,7]. So, in this research work, the Genetic approach is used for finding the optimal loading of bins in to the container by satisfying most of the practical constraints [3].

GA uses a random approach [2] and bin packing needs an ordered packing approach, so it needs a separate Tuning algorithm in conjunction with the GA for ordered feasible packing, from the optimal result obtained by the random genetic approach. The aim of this tuning algorithm in this paper is to partition the items between bins in an ordered fashion, so that the maximum number of bins without an empty volume can be packed in the container. In general, the use of Hybrid GA in combination with the tuning algorithm is to optimize the 3D bin packing problem.

V. GENETIC ALGORITHM

Recent revolutions in molecular genetics made clear that the modular organization of genes is highly important for the evolution of complexity. The Evolutionary concept of GA was introduced by John Holland in 1975 at the University of Michigan [1]. GA is a search procedure based on the mechanism of natural selection and genetics, successfully applied to variety of complex multi variable optimization problems for getting an optimal and feasible solution from a large space of population without specific structures or prior information[2]. Various stages of the genetic algorithm are explained in the following sections.

A. Population Generation

First stage in using the Genetic Algorithm is an initial population generation with 'n' number of strings generated randomly, based on probability logics [1, 2]. In this research work, each chromosome segment represents a bin. Normally, a better solution is obtained as the population size increases, with an increase in computational time as a result, there exists a trade off between both. Population size depends on complexity and number of variables involved in the application [1] and no hard rule exists so far. In this work, population size was set at 100 with 150 genes in a chromosome generated using a randomize function, based on the experimental results obtained for various population sizes from 10 to 150. Population size of 10 leads to more number of generations and the result doesn't give much difference beyond the population size of 100. Number of box types taken for experimental implementation was four i.e. the decimal numbers 1,2,3,4 represent a cube, a rectangular prism, a cylinder[13] and a sphere respectively. The Sample generated chromosome is shown in figure 1. Each number in that Figure 1 represents a bin type[14]. For example, from Figure 1, the First digit '4' represents the first sphere in the database, Next digit '2' represents the first rectangular prism, next digit '2' represents the second rectangular prism in the database, and so on. Similarly, 100 chromosomes with 150 genes generated using randomize function represent a 100 set combinations of 150 bins.

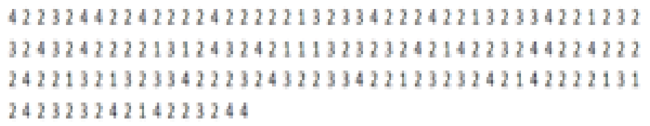

4 2 2 3 2 4 4 2 2 4 2 2 2 2 4 2 2 2 2 2 1 3 2 3 3 4 2 2 2 4 2 2 1 3 2 3 3 4 2 2 1 2 3 2
3 2 4 3 2 4 2 2 2 2 1 3 1 2 4 3 2 4 2 1 1 1 3 2 3 2 3 2 4 2 1 4 2 2 3 2 4 4 2 2 4 2 2 2
2 4 2 2 1 3 2 1 3 2 3 3 4 2 2 2 3 2 4 3 2 2 3 3 4 2 2 1 2 3 2 3 2 4 2 1 4 2 2 2 2 1 3 1
2 4 2 3 2 3 2 4 2 1 4 2 2 3 2 4 4

Figure 1: Sample Chromosome

B. Reproduction

The Second stage in GA is the reproduction or crossover between parents in the population [1]. To achieve a good loading pattern that maximizes the volume utilization of a container, it is necessary to select the best two parents from the population, based on the fitness function value [2]. The formulation of the fitness function will be discussed later in section 5.5. Parents for the crossover and crossover site were selected from the population, using random method. In general, the Crossover probability can be 80 % and it will vary from application to application. In this research work, it was set to 200 % which leads to faster convergence towards the best solution. Double two point cross over implies that every parent in the population has to make a cross over with two other parents. So, one of the generated offspring inherits the best properties from both the sets of parents. Packing and positioning of bins found, may not be optimal in the first generation itself. But this type of inheriting the best properties from more than two parents, will lead to faster convergence to the best optimal solution for packing boxes in to the container.

As the parent size in this work was set to 150, the single point crossover didn't generate much difference in the fitness function value. In other words, the GA didn't search in larger space as the search area gets shrunk. In order to widen the search space, the two point Crossover operator was applied between two randomly selected parents and allowed to reproduces two offsprings. The two offsprings generated as a result of two point double crossover are shown in figure 2. From figure 2, It is seen that 'temp offspring' is generated from 'parent 1' and 'parent 2' with the crossover site of 57 and 102. Similarly 'offspring 1' obtained from 'parent 3' and 'temp offspring' with random crossover site of 30 and 75.

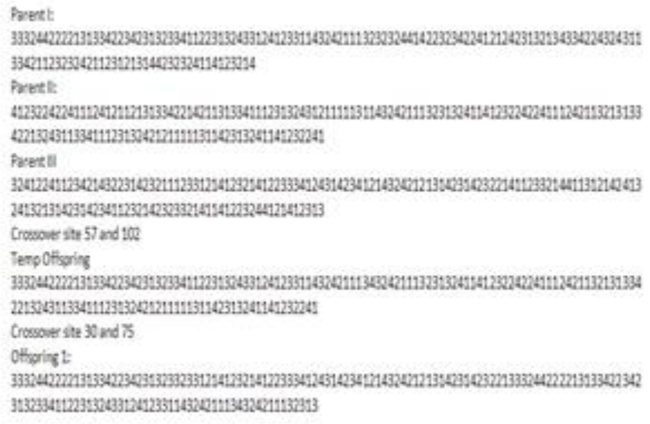

Parent I:
33324422221313342234231323341122313243312412331143242111323232441422323422412124231321343342243243 11
3342112323242112312131442323241141232 14
Parent II:
41232242241112412112131334221421131334111231324312111113114324211132313241141232242241112421132131 33
42213243113341112313242121111131142313241141232241
Parent III
32412241123421432231423211123312141232141223334124314234121432421213142314232214112332144113121424 13
24132131423142341123214232332141541223244121412313
Crossover site 57 and 102
Temp Offspring
33324422221313342234231323341122313243312412331143242111343242111323132414123224224111242113213133 4
22132431133411123132421211111311423132411412322 41
Crossover site 30 and 75
Offspring 1:
33324422221313342234231323323312141232141223334124314234121432421213142314232213332442222131334223 42
31323341122313243312412331143242111343242111323 13

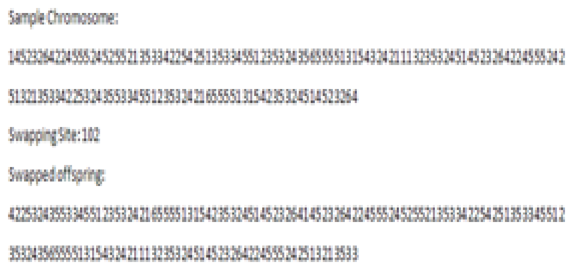
Figure 2: Crossover

C. Mutation

Sometimes, the Fitness function value of the obtained offsprings may seem to stagnate around the optimal point[1]. This problem can be solved by applying the mutation operator, which helps to attain the best optimal point. Mutation is the operation of swapping the individual gene or string in a chromosome by selecting the position of string in the chromosome randomly[2]. In this research work, a two point mutation is used only to retain the best property of the offspring. Mutation points are selected at random. As a special heuristic in this work, the Mutation operator is used to change the orientation of the bin between the length, breadth, and width (any two parameters at random will be swapped), while the mutation operates over it and in turn the box orientation gets changed. Changing the bin orientation helps a lot for optimal packing. Left side of the database gives the details of the bins entered by the user. Newly generated offspring is given in the centre column. If the Mutation site generated at random was 9, then length and width of the 9th bin in the database was swapped. Thus the orientation of the bin is changed and for this dimension, the fitness function optimizes the packing volume.

D. Swapping

Swapping is a special type of GA operator used in this research, to improve the search space instead of being focused towards a particular local optima [2]. In this research, a single point random swapping was implemented to expand the search space. Swapping site has to be selected at random and strings beyond the swapping point should swap to the front. Probability of swapping was set to 100%. By doing many simulations, it was found that the swapping avoids the repetition of parents of the same fitness function value, or in other words, it avoids the stagnation of GA at a point to generate the best mix of different boxes which occupy the maximum volume inside the container. Figure 3 explains a sample swapping operation. Randomly generated swapping site was 102 then strings beyond 102 were swapped to the front and shown in swapped offspring in figure 3.



Sample Chromosome:

14523264224555245255213533422542513533455123532435655551315432421113235324514523264224555242

5132135334225324355334551235324216555513154235324514523264

Swapping Site: 102

Swapped offspring:

42253243553345512353242165555131542353245145232641452326422455524525521353342254251353345512

3532435655551315432421113235324514523264224555242513213533

Figure 3 : Mutation

E. Fitness Function

The Fitness function for a single container loading problem [6] can be formulated as in equation 1.

$$Min.\ F(x) = (Container - Packed\ box\ )Volume \quad (1)$$

where F(x) is a minimization fitness function used to calculate the survival value of the chromosomes. The Fitness function is the minimization function of the difference between the container volume and sum of volume of all the packed boxes[8]. Fitness function value ranges from 0, which denotes complete packing without empty space inside the container and an increase in the fitness function value denotes a proportional increase in the empty volume inside the container. Fitness function was developed for maximizing the container volume and minimizing the empty space inside the container. By conducting various experiments, Results obtained from the minimization function were better than those of the maximization function for this container loading application. V(x) in equation 1 is the volume function which calculates the total volume occupied by boxes inside the container. Using this function it is able to calculate the used and free space.

F. Termination Criteria

First generation terminates with calculating of the fitness value. The obtained offsprings and their fitness values were tested with the termination conditions. In each generation, new offsprings were reproduced and those offsprings have to replace the weak chromosomes present in the parent population with less fitness value[1]. The best 50 % (75 chromosomes) of offsprings has to replace the 50 % worst parents and a new generation produced with 75 best parents and 75 best children to form a population of the best and the same procedure has to be repeated the until the optimal result is obtained. Finally, the generated chromosome has an optimal and feasible solution for packing the boxes into the container [1]. Termination criteria will vary from application to application. In this research, four basic termination conditions were set, based on the results obtained after conducting and analyzing many experiments.

VI. EXPERIMENTAL IMPLEMENTATION

Once this model was decided upon, implementation was done using the Visual Basic (VB) software with MS-Access as the database. Experiments were conducted using Intel® Pentium (IV) with 2.06 GHz CPU and 1 GB RAM. A number of different modules of classes and objects were developed in VB to model a genetic algorithm for solving the container loading problem. Various modules involved in this work are explained in the following sections.

A. Input Module

The GUI for user input was developed in such a way that the user can enter the required box dimensions[16] through the keyboard or select from the database. The primary/essential parameters for bins were shape, along with length, width and breadth. Secondary parameters were weight and constraints. Once the user enters the above mentioned details, the developed module will assign a unique code for each box for further identification and analysis, based on the entered value. Unfilled details of the box are filled by default values generated from the expert system module.

B. Genetic Algorithm module

Stored data will be combined to make up one chromosome of length 150. In GA several control parameters need to be set appropriately to optimize its performance [1] and the GA configures a few variables after conducting several preliminary experiments to determine the suitable parameter values and those values shown in table 1.

Table 1: Genetic Parameters

Parameter	Value
Crossover rate	200 %
Crossover point	2
Mutation rate	200 %
Mutation point	2
Population size	100
Max no of generation	100
Genome length	150
Min. fitness value	0 - 10

An initial population of size 100 and chromosome length of 150 was generated randomly [3]. Chromosomes were allowed to operate over the fitness function and the calculated fitness values should be stored again in the same database for sorting from the best to the worst. Random methods were used for selecting the parents in this work for reproduction. In the reproduction stage, two point double crossover operators were applied and the crossover points in the chromosomes were selected again at random [1] for faster convergence. At the crossover point, strings were exchanged between the two parents and as the result, two offsprings were generated. The generated offspring may stagnate near the optimal point and to avoid this, the two point mutation operator was applied over the offspring. Mutation was northing but the swapping of a particular gene at a random position, so that the orientation of that box was swapped i.e. length to breadth and vise versa. On the other a hand higher mutation rate deviates from the optimal point [2], so in this work, the mutation rate was selected in such a way that only one swapping per parent was done. The single point swapping operator was also applied to avoid repetition in the fitness function values and to be avoid focusing towards a local minimum optimal point. The fitness values for each and every generated individual offspring (chromosome) in a population were evaluated and the offspring with better fitness values has to be added to the initial population by replacing the least fitness value parent chromosomes.

Thus the first generation was completed with a new set of the best parents for the next generation. For the second generation, the same procedure was repeated and it continues till the number of generations reaches 100 iterations, or continuous generations of offspring with the same character are generated [4]. The final offspring gives the best placement of the mix of boxes inside the container, the obtained result may not be the optimal result, but it will be the best feasible result [5].

C. Heuristic Tuning Module

All the parameters and operations from the initial population generation till the termination, which were involved in the Genetic approach were operated using the randomize function. On the other hand, the packing of bins into the container is purely an ordered approach. So it becomes very clear that the output obtained from the random approach can not be used directly for an ordered application. GAs may require a long execution time in order to find a good packing solution for optimizing both the volume and sequence so in this work, the GA is used to optimize the packing volume and a tuning algorithm is used to pack the boxes in to the container. By using a hybrid GA[10], the execution time needed to find good solutions is reduced and the GA result synchronizes with the application. the Tuning algorithm decodes the GA output into a packing sequence, thereby forming a hybrid genetic approach to tackle the container loading problem. These hybrid approaches not only fill the gap, but also bring the GA output in a layman-understandable format. Some of the criteria to pack the boxes inside the container in this research work are (i) The container origin should be the lower most left corner (0, 0, and 0) and also placement coordinate for the first box [11]. (ii) The length, width and height of the container should be along the x , y and z-axis respectively [3]. (iii) The container is placed in the Cartesian coordinate system to be able to relate the placement of the different boxes to one another. (iv) the layer by layer filling concept is used in this work. The fist box to be placed has the placement coordinate as (0, 0, 0) i.e. the box should be placed at the lower left most corner of the container. Once the base box is placed, then the placement coordinate is shifted to the opposite corners of that placed box and the process continues. These calculated coordinate values are shown in table 2. Second column in table 2 represents the GA output and the next columns represent the box types and dimensions. The seventh column represents the unique code generated by this module. The base origin column represents the placement corner for each and every bin in the Cartesian coordinate system. The diagonal corner values in the table 2 represent diagonal corner of each and every bin which is used to avoid and to check the overlapping of bins. The 'Layer' column in the table represents the placement of boxes in the XY plane. 'Column' in table represents the bin placement along the Y axis and the 'Row' column represents the bin placement along X axis. So this format helps the user to understand and visualize the bin packing inside the container easily and also it helps to find the placement corners and the amount of gap formed in each layer.

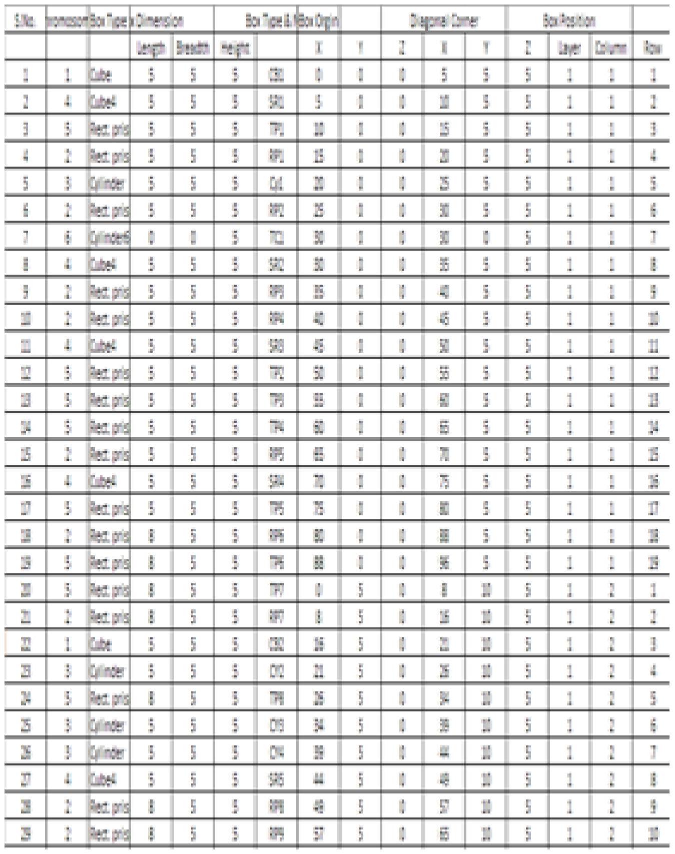
Once an empty volume is found, it can be filled by suitable boxes to achieve complete packing inside the container [15] and thereby obtaining the minimum empty space using the tuning algorithm. Plan view of a layer for table 2 is shown in figure 4.

The left and right side views shows the top view of packing of the bin for the data shown in table 2, before and after applying Tuning algorithm respectively. Because of the arbitrary sizes and heterogeneous shapes, some times a gap or space may be formed in layer packing. This can be avoided by developing a heuristic tuning algorithm.

D. Output Module

The positions of all the bins in the Cartesian coordinate system were given by the software. The placing arrangement of the bins inside the container in the layer by layer format was also given in the output in a graphical format.

Table 2: Output Format



S.No.	hromosom	Box Type	Dimension			Box Type & N	Box Origin			Diagonal Corner			Box Position		
			Length	Breadth	Height		X	Y	Z	X	Y	Z	Layer	Column	Row
1	1	Cube	5	5	5	CB1	0	0	0	5	5	5	1	1	1
2	4	Cube4	5	5	5	SR1	5	0	0	10	5	5	1	1	2
3	5	Rect. pris	5	5	5	TP1	10	0	0	15	5	5	1	1	3
4	2	Rect. pris	5	5	5	RP1	15	0	0	20	5	5	1	1	4
5	3	Cylinder	5	5	5	CY1	20	0	0	25	5	5	1	1	5
6	2	Rect. pris	5	5	5	RP2	25	0	0	30	5	5	1	1	6
7	6	Cylinder6	0	0	5	TC1	30	0	0	30	0	5	1	1	7
8	4	Cube4	5	5	5	SR2	30	0	0	35	5	5	1	1	8
9	2	Rect. pris	5	5	5	RP3	35	0	0	40	5	5	1	1	9
10	2	Rect. pris	5	5	5	RP4	40	0	0	45	5	5	1	1	10
11	4	Cube4	5	5	5	SR3	45	0	0	50	5	5	1	1	11
12	5	Rect. pris	5	5	5	TP2	50	0	0	55	5	5	1	1	12
13	5	Rect. pris	5	5	5	TP3	55	0	0	60	5	5	1	1	13
14	5	Rect. pris	5	5	5	TP4	60	0	0	65	5	5	1	1	14
15	2	Rect. pris	5	5	5	RP5	65	0	0	70	5	5	1	1	15
16	4	Cube4	5	5	5	SR4	70	0	0	75	5	5	1	1	16
17	5	Rect. pris	5	5	5	TP5	75	0	0	80	5	5	1	1	17
18	2	Rect. pris	8	5	5	RP6	80	0	0	88	5	5	1	1	18
19	5	Rect. pris	8	5	5	TP6	88	0	0	96	5	5	1	1	19
20	5	Rect. pris	8	5	5	TP7	0	5	0	8	10	5	1	2	1
21	2	Rect. pris	8	5	5	RP7	8	5	0	16	10	5	1	2	2
22	1	Cube	5	5	5	CB2	16	5	0	21	10	5	1	2	3
23	3	Cylinder	5	5	5	CY2	21	5	0	26	10	5	1	2	4
24	5	Rect. pris	8	5	5	TP8	26	5	0	34	10	5	1	2	5
25	3	Cylinder	5	5	5	CY3	34	5	0	39	10	5	1	2	6
26	3	Cylinder	5	5	5	CY4	39	5	0	44	10	5	1	2	7
27	4	Cube4	5	5	5	SR5	44	5	0	49	10	5	1	2	8
28	2	Rect. pris	8	5	5	RP8	49	5	0	57	10	5	1	2	9
29	2	Rect. pris	8	5	5	RP9	57	5	0	65	10	5	1	2	10

Details of the packed and unpacked boxes were also obtained. All the results were stored in an MS excel/Access database for further analysis.

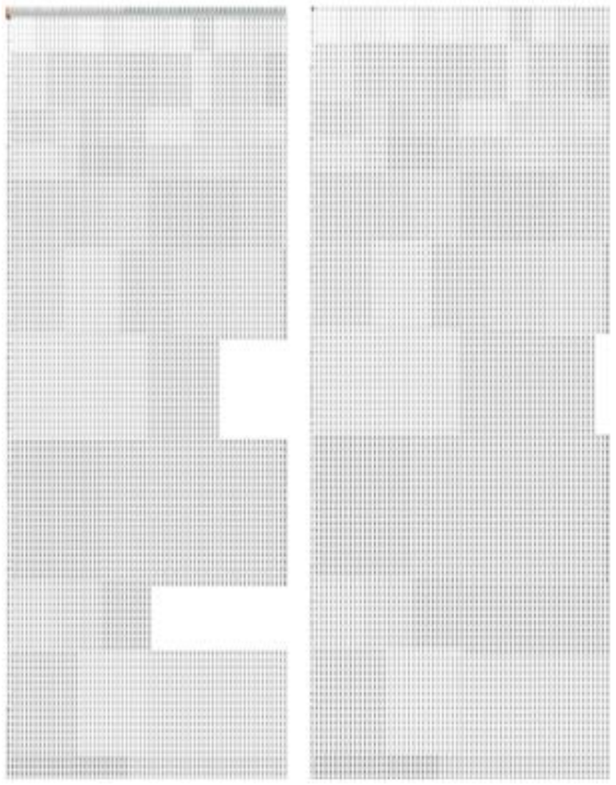


Figure 4: Plane View of before and After applying Tuning Algorithm

VIII. RESULT

In this work, it was found that the computational time of container loading with the two point double crossover and orientation mutation GA was improved, when compared to the traditional optimization/GA techniques and the results quite practically acceptable. To improve time, various test cases were taken for consideration by varying the population size, parent size, cross over probability and mutation probability to conduct experiments to test for factors that affect computation time of the loading process. Finally, the GA was designed with effective operators, an appropriate selection criterion, and adequate population size for each operator as mentioned earlier. In this research work most of the practical constraints were considered and the results obtained by a combination of the hybrid GA with the Tuning algorithm were found satisfactory and feasible for packing, when compared to other standard search algorithms. No boxes remained unloaded using this algorithm will helps in validating the developed heuristic genetic algorithm.

CONCLUSION

The data analyzed so far with this research supports the fact, that a two point double crossover, orientation mutation operator and swapping GA will decrease the execution time of the container loading problems by satisfying the practical constraints and give a feasible real time packing pattern. As the GA is a random approach and packing application is an ordered approach, it needs a heuristic tuning algorithm to combine with the GA. Sorting the user input bin data will help to satisfy the weight and stability constraints.

FUTURE SCOPE

Further research may attempt to improve this algorithm, by a more appropriate multi container sequence and initial loading. A different initial loading position may be studied to improve the optimal loading position and volume utilization. Further research may also include special restrictions that specifically exist in other container loading applications, like air cargo loading (Wing packing), because small gaps between the boxes will not be allowed in order to avoid inertia that exists during take off. Other research may also focus on non-regular shapes of the container. The algorithms overall behavior is further examined by looking at its convergence over time.

References

[1]. D.E. Goldberg, “Genetic Algorithm in Search, Optimization and Machine Learning”, Addison Wesley, 1988.
[2]. M.Mitchell, “An Introduction to Genetic Algorithm”, MIT Press, 1996.
[3]. E.Hopper and B. Turton, “Application of Genetic Algorithm to Packing Problems – A Review”, Springer Verlag, London, pp 279 -288, 1997.
[4]. KA Dowsland, EA Herbet, “Using Tree Search Bounds To Enhance A Genetic Algorithm Approach Two Rectangle Packing Problems”, European Journal of Operation Research 168, 390-402, 2006.
[5]. R. Korf, “An Improved Algorithm for Optimal Bin Packing”, In proc. IJCAI, pages 1252 – 1258, 2003.
[6]. D.Pisinger, “Heuristic for Container Loading Problem”, European Journal of Operation Research 141, 292 – 382, 2002.
[7]. H.Gehring, A. Bortfeldt, “A Genetic Algorithm for Solving the Container Loading Problem”, International transactions in Operation Research, 44, pages 401 – 418, 1997.
[8]. S. Martello, D. Pisinger, “The Three Dimensional Bin Packing Problem”, Operation research 48, 256-267, 2000.
[9]. EE Bischoff, “Three Dimensional Packing of Items with Limited Load Bearing Strength”, European Journal of Operation Research 168, pages 952-966, 2004.
[10]. A. Bortfeldt and H. Gehring, “A Hybrid Genetic Algorithm for Container Loading Problem”, European Journal of Operation Research 131, page 143 -161, 2001.
[11]. SG. Christensen and D.M. Rousoe, “Container Loading with Multidrop Constraint”, masters thesis, Informatics and mathematical Modelling, Technical University of Denmark, DTU, Lyngby, 2007. http://www2.imm.dtu.dk/pubdb/p.php?5225.
[12]. A.P. Davies and E.E. Bischoff, “Weight Distribution Considerations In Container Loading” European Journal of Operation Research 114, pages 509-527, 1999.
[13]. John A George and Jennifer M George, “Packing Different Sized Circles into a Rectangular Container”, European Journal of Operation Research 84, pages 693-712, 1995.
[14]. E.K. Burke, M.R. Hyde, “Evolving Bin Packing Heuristic with Genetic Programming”, School of computer science and information technology”, UK. http://cs.nott.ac.uk/~mvh.
[15]. Dyckhoff H. A typology of cutting and packing problems. European Journal of Operation Research 1990;44:145–59.
[16]. Kang J, Park S. Algorithms for the variable sized bin packing problem. Eur J Oper Res 2003;147:365–72.