Journal of Advanced Computer Science & Technology, 4 (1) (2015) 190-197
www.sciencepubco.com/index.php/JACST
©Science Publishing Corporation
doi: 10.14419/jacst.v4i1.4348
Research Paper

Problem solving of container loading using genetic algorithm based on modified random keys

Mohammad Sadegh Arefi *, Hassan Rezaei

Department of Computer Science, Faculty of Mathematics, University of Sistan and Baluchestan, Iran
**Corresponding author E-mail: Arefi.Sadegh@yahoo.com*

Abstract

This article presents a solution to the container loading problem. Container loading problem deals with how to put the cube boxes with different sizes in a container. Our proposed method is based on a particular kind of genetic algorithm based on biased random keys. In the proposed algorithm, we will face generations' extinction. Population decreases with time and with the staircase changes in the rate of elitism, the algorithm is guided towards the global optimum. Biased random keys in the proposed method are provided as discrete. The algorithm also provides the chromosomes that store more than one ability. In order to solve container loading using a placement strategy, due to the size of the boxes and containers, the containers are classified as small units and equal unites in size. Finally the algorithm presented in this paper was compared with three other methods that are based on evolutionary algorithms. The results show that the proposed algorithm has better performance in terms of results and performance time in relation to other methods.

Keywords: *Container Loading; Evolutionary Algorithms; Genetic Algorithm; Random Keys.*

1. Introduction

Container loading problem simply stated as how to fill a container with items and goods that are in the cube's dimensions. According to the transportation system in economic and business worlds that much is done through containers, their loading has been analyzed in different perspectives. This is a type of cutting and packing problems [8].
The issue is also among NP-hard problems [4]. The larger the container space and the more the items are the more difficult and time-consuming the find of answers will be. Finding a sequence of boxes that are entered arranged into a container and how to place the boxes in the interior of the container will be complex. Solving this problem by deterministic algorithms cannot be in reasonable time. Determination of the input entrance sequence into the container and their placement are the key issues to be addressed in container loading problems.
Given the importance of the container loading problem, its solution has a long history. George Robinson in 1980 [7] presents a layer algorithm for filling the container and smoothed the way for others. In 1990 Daykof typology categorized cutting and packing problems based on the dimension number, font characteristics, constraints and purpose [8]. So far, this problem has been investigated in different ways, such as branch and bound [6], Tabu search [7], genetic algorithms [1] and [2], Smart safety system algorithms [9], and Particle swarm optimization [9].
In this paper we, took the idea of encoding scheme based on random key of Bean in 1994 [14], explained the introduction of an improved genetic algorithm based on random keys. Biased random keys in the proposed method are provided as discrete. The chromosomes in these algorithms store the ability of the sequence boxes and the presence in each step of box loading. Also, given that the state space of the problem is very large, it is recommended to use the relatively large population at first and then decrease the population over the generations. Through this process, we try to concentrate on the best algorithm. The rate of elitism in this algorithm is not constant. The rate begins with fewer amounts in early generations, and also increment of the rate will increase the quality of answers.
The paper has been organized as follows: In section II we define the problem. Then, in the third section discusses the methodology. The placement of the revised strategy and the proposed genetic algorithm proposed to solve the issue is

discussed at the end of this section. The fourth section presents the results, and the conclusion is presented in the fifth section.

2. Basic concepts

2.1. Definition of container loading problem

Container loading problem deals with how to put the cube boxes with different sizes in a container. In this paper, we describe a comprehensive overview of the problem. In other words, the items in this issue have been assumed regardless of their type, but just the cube and container size in standard mode. Solving this problem can be divided into two basic parts:
1) Selecting the box input sequence to the container.
2) Management of the interior part of the container and the layout of the boxes inside the container.
The first part of the problem is to check the boxes differentness sequences into a container. In this process, the boxes are in a priority sequence and it determined that, which box should be marked as the first box into the container, and also the next, etc. The number of states exists, depending on the variety of the items and the total number of boxes that can be used to produce different sequences. For example, suppose there are 10 different types of boxes, 2 boxes of each type, total number of sequences is:

$$\frac{20!}{2^{10}} = 237{,}588{,}086{,}7360{,}000$$

As it is known, the result is a very large number. Now assume that the number of box species and the number of boxes are 100 and 200, respectively. What number we will face with? Assessment of all the states in the traditional way is impossible. So the right sequence should be found without investigating other ways. Many researches have been done in this area, such as branch and bound method presented by Martello and colleagues in 2000 [6], or Tabu Search methods by Bortfeldt and Gehring [15]. Peeraya et al had been resolved the issue in three methods of evolutionary algorithms [9]. Gonçalves and Resende improved the problem with the idea of biased random keys genetic algorithm (BRKGA) [1] and [2].
The second part of the problem is about how the boxes layout in three-dimensional space. In this section, the boxes are arranged in the container with respect to the sequence of the first part. The arrangement can be done in various ways. A set of processes to sort and manage the complex processes inside the container is called placement strategy. This strategy is defined based on different ideas and problem types. Among the most prominent methods are the methods of placement strategy by George and Robinson [7]. The strategy is to build a layer in the container to manage the interior space. Or Bischoff and Ratcliff method can be used which is based on the stack creation [16]. Eley in 2002 provided a method based on the building blocks of the boxes to solve this problem [13]. Gonçalves and Resende, using ideas of Lai and Chan [10] provided an efficient way about the analysis of container interiors.

3. Methodology

In this section we offer suggestions for solving the container loading problem. Using placement strategy proposed by Gonçalves and Resende [1], a higher speed has been achieved for proposed divides in the interior container. Furthermore, using genetic algorithm based on the Bean random key method [14] and the idea of a genetic algorithm introduced by Gonçalves and Resende [1], we introduced a genetic algorithm to act based on discrete random keys. In this algorithm, different population is used. More details of the proposed method are in the following.

3.1. Placement strategy

The proposed placement strategy is based on the method proposed by Gonçalves and Resende [1]. This is one of the most accurate methods of loading process. The smallest of spaces can be examined in the method. It generally consists of four parts: identified maximal spaces, prioritized maximal space, creating layers of boxes and joining maximal-spaces. When a box or a set of boxes are determined to be located in the container, the boxes are placed in a container based on the strategy and the data can be updated. In this method, the interior of the container is classified as defined by the maximal space of Lai and Chan [10]. When the entrance of a box to the container is selected, the system is built upon layers of boxes, fills selected maximal space. The maximal space inside the container can be detected again. This method takes the maximal sharing spaces into consideration. Gonçalves and Resende placement strategy has very high accuracy. But this will reduce execution speed. The system also has the same behavior with different types and sizes of boxes. In other words, if the collection of boxes that we are dealing with is very small compared to the size of the container, the method will be very slow, since interior space is re-arranged after every entrance of a series of boxes in a container. So, for compact sets the process will be time consuming.

Our recommendation to improve Gonçalves and Resende placement strategy is to divide the interior container based on boxes' volume. In other words, if the container is too small compared to existing boxes, its space is divided into smaller parts. It can be prevented further processing areas. The idea of longitudinal and transverse incision in the container will become possible. Fig 1 is a representation of this division.

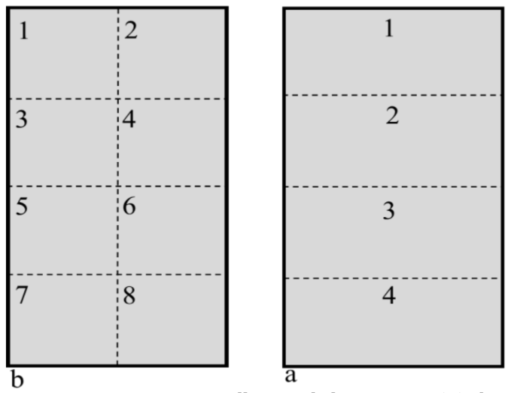


Fig. 1: In Figure (A) the Container by Three Transects in Tetra- Equally Divided. In Figure (B) the Container Cut by Three Transverse and One Longitudinal Section Is Divided Into Eight Equal Parts.

3.2. The proposed genetic algorithm

The genetic algorithm is an evolutionary algorithm of a set of algorithms. This algorithm is based on Darwin's evolution theory, a path the best will remain and the poor will extinct. Holland (1975) was the first who proposed the theory of genetic algorithms. This algorithm is widely used today in all the sciences. This is a very efficient algorithm for solving the container loading problems. One of the genetic algorithms introduced in problem solving is Gonçalves and Resende genetic algorithm. They used biased random keys [14] to introduce the algorithm which is able to create different sequence of boxes.
In this paper, we used the idea of a genetic algorithm based on random keys and Gonçalves and Resende proposed genetic algorithms to propose Modified BRKGA (MBRKGA) in which random keys follow a new structure. These keys are defined in the form of discrete intervals instead of continuous space between zero and one. This will avoid the creation of similar incidences in a population. In this algorithm, we provide a new mechanism for solving the container loading problem. Due to the variability of states in the problem, the algorithm used extremely large population size at first. Fewer populations cause the concentrations on the best. The elitism rate of this algorithm is not consonant. The rate is little in the early generations, and also increases by the increment in the quality of answers and finally reduced in the last generation. This reduction is in line with the focus on the best final answers and is used to find new solution. A new algorithm with an intersection process was introduced that parents are selected using a roulette wheel based ranking [17]. Following is an introduction of the algorithm described above.

3.2.1. Coding

In the proposed approach, each answer length is equal to the number of items in each box and each gene contains a number with the integer and decimal parts. The integer part is a number between one and one hundred. This number specifies the number of remaining boxes that can be used at any stage of positioning. At least this number depends on the number of boxes that are supposed to be loaded, this number will be determined according to Table 1. This idea is based on the principle that any increase in box items would reduce the number of each type and conversely, decreasing the box items will increase the number of each type. Fractional part of each gene is the same random key that is used in BRKGA algorithm [14]. The key is obtained by any change based on 1 and 2. This shift is for purposeful random keys. This number specifies the priority of boxes.
The answers will be composed in two functions. The first feature is that the amount of container that can be loaded at any stage and the second feature is the boxes that are prioritized. Fig 2 displays a chromosome with 10 boxes.

$$\text{bias} = \frac{1}{5\times \text{k}} \tag{1}$$

$$\text{key(i)} = \text{round down(rand } \times 5 \times \text{k)} \times \text{bias} \tag{2}$$

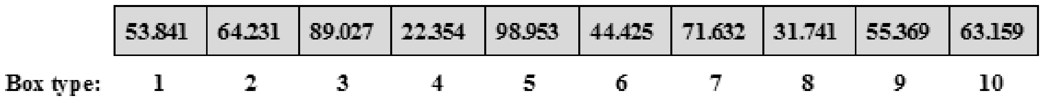
Where k is the number of boxes, and key (i) is the i-th gene random key.

Table 1: Specify the Minimum Percentage of the Presence of the Boxes in the Downloads.

The minimum percent attendance for any box	The number of such boxes
20	2-10
30	11-20
40	21-30
50	31-40
70	41-50
100	More than 50

3.2.2. Decoding

Decoding process of the algorithm consists of two steps. First, the process is the common of evolutionary algorithms. The second stage can be seen as the interface between evolutionary algorithm and placement system. This process is to update the decoded sequences in the first stage. Through this process, every step of the sequence box placement has different priorities. Fig 3 displays the structure of the proposed decoding process. The stages of this process are as follows.



	53.841	64.231	89.027	22.354	98.953	44.425	71.632	31.741	55.369	63.159
Box type:	1	2	3	4	5	6	7	8	9	10

Fig. 2: View A Chromosome for 10 Different Types of Boxes.

First stage: In this stage of evolutionary algorithm answers receives, is decoded and then deposit to the placement algorithm. The answer can be arranged according to the decimals and boxes with smaller random number are allocated to the highest priority. Then, based on the integer part of each element in the solution, the percentage of the remaining boxes that can participate in every stage of the process of loading is determined.
Second stage: This stage is an iterative process that repeatedly attempted to change the priority of boxes. This produces a non-uniform sequence of boxes. At this stage, after the loading process with a box with higher priority, this is to change the decimal number corresponding element of box in the answer which is done by the following equation.

$$X = x_1, x_2, \dots \dots \dots, x_K \tag{3}$$

$$X = \begin{cases} X\, x_C + \text{bias} < 1 \\ \sum_{i=1}^{k}(x_i - \min_{i=1}^{k} x_i\ ) \text{ else} \end{cases} \tag{4}$$

$$x_C = x_C + \text{bias} \tag{5}$$

$$t_C = \text{Round up } (\frac{z_C}{100} \times T_C) \tag{6}$$

Where x_i is fractional part corresponding with box type i in the answer, c is the box that locates at priority load, x_C is the fractional part of the box that has a higher priority, T_C is total number of remainder boxes type c, t_C is the maximum number of type c boxes that can be loaded at the present stage, k is the number of items in the boxes, X is x_i vector and bias is obtained from Equation 1. In this process, the relationship between Eq.4 and 5 is executed. Then this answer can be sorted based on floating-point numbers and identified the priorities and delivered to replacement system.
Equation 6 specifies maximum number of boxes that can be loaded at any stage. This process continues until the end of the loading process.
Fig 4 displays the decoded answer in 10 boxes. This figure displays four-step update priorities of boxes.

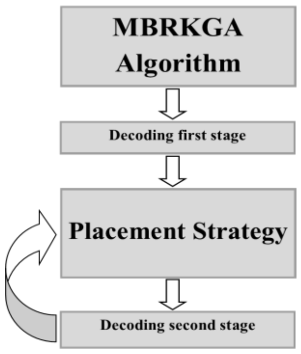
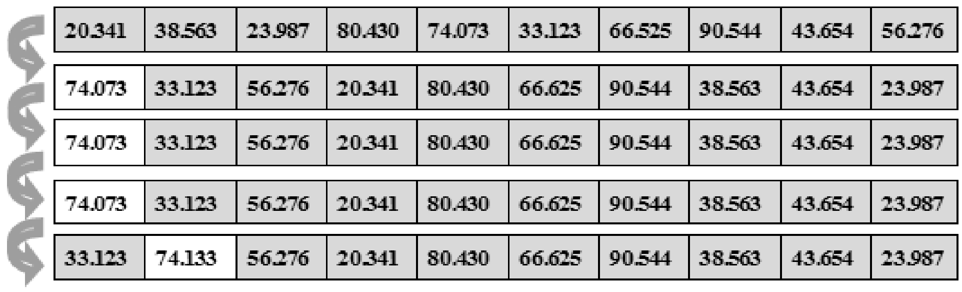


Fig. 3: View Decoding Process.

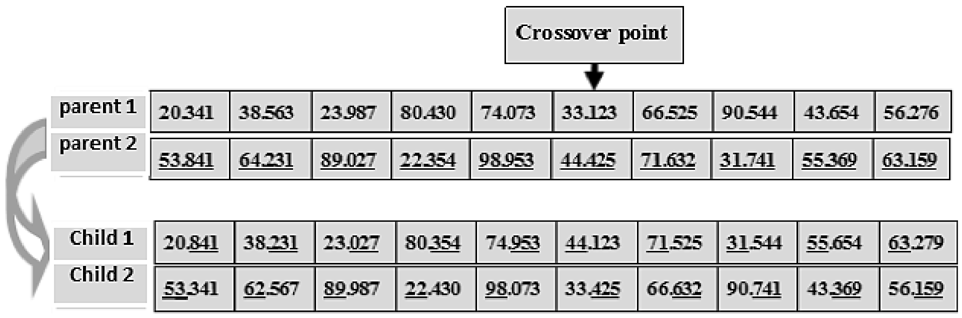


20.341	38.563	23.987	80.430	74.073	33.123	66.525	90.544	43.654	56.276
74.073	33.123	56.276	20.341	80.430	66.625	90.544	38.563	43.654	23.987
74.073	33.123	56.276	20.341	80.430	66.625	90.544	38.563	43.654	23.987
74.073	33.123	56.276	20.341	80.430	66.625	90.544	38.563	43.654	23.987
33.123	74.133	56.276	20.341	80.430	66.625	90.544	38.563	43.654	23.987

Fig. 4: Show Decoded A Chromosome with the Ten Type Box after Four Steps.

3.2.3. Crossover operator

Two chromosomes are selected at intersection process through the roulette wheel method based rankings. First chromosome is selected among elites and the second chromosome from the population. In this method, the two chromosomes are produced as a child. This method is similar to a single point intersection. In this method, a point is selected on the chromosomes, and then the child chromosomes inherit the right section before crossing point from one parent and the right section after crossing point from the other. The decimal part of the children have been determined likewise. There is a point in the intersection that the decimal and integer part of each child for a parent are not selected simultaneously. In other word, the movement of genes in each sector is in the cross. Note in Fig 5.



parent 1	20.341	38.563	23.987	80.430	74.073	33.123	66.525	90.544	43.654	56.276
parent 2	53.841	64.231	89.027	22.354	98.953	44.425	71.632	31.741	55.369	63.159
Child 1	20.841	38.231	23.027	80.354	74.953	44.123	71.525	31.544	55.654	63.279
Child 2	53.341	62.567	89.987	22.430	98.073	33.425	66.632	90.741	43.369	56.159

Fig. 5: View Crossover Operation between the Two Chromosomes.

3.2.4. Mutation operator

Mutation Operator is a common method in this algorithm. The method determines the rate of mutation to identify a number of genes in the whole population (except for the top element of the population). Then, these genes are replaced with randomly generated genes. So the correct decimal and integer of each gene specified based on the model presented in the encoding pattern.

3.2.5. Fitness function

Each chromosome evaluation can be done according to Equation 7.

$$\text{fitness}(c) = 100 \times \left(1 - \frac{\sum_{i=1}^{k} t_i \times v_i}{V}\right) \tag{7}$$

In this equation, c is the target chromosome, k is the number of different types of boxes, t_i is total number of boxes type i in a container, v_i is the volume of box type i and V is the volume of the container. The purpose of this algorithm is to find the minimum.

3.3. Problem solving

Being familiar with the placement strategy and structure of MBRKGA algorithm, we discuss how to solve the problem; Fig 6 shows an overview of the problem. In this method, a large population is generated using of random coding. Then the population is decoded. In the decoding process, each chromosome is decoded at the first stage; then the decoded chromosome is deposited to the placement system. In the system, the boxes with higher priority are selected at first based on the received sequence, then the maximal space with higher priority is selected, then the appropriate layer of boxes is created for placement. Next, the layers lie in the maximal space and new maximal space is identified. Maximal spaces are updated and regulated in the following. After a placement strategy implementation stage, the second stage decoding turns out. In this section, boxes priority is updated. After the decoding process, chromosomes are evaluated. Population size and selection rate change in staircase form according to the mentioned pattern after evaluating all of the population. The model is specified in Table 2. This cycle continues until a stop condition. Stop condition of the algorithm is to achieve a desired value to produce a given number of generations.

Table 2: Changes in Population Size and Selection Rate in Genetic Algorithms.

selection rate	Population size	Number of generations	Generation number
0.20	400	2	1-2
0.25	300	3	3-5
0.30	200	5	6-10
0.25	100	20	11-30
0.20	50	70	31-100

4. Experimental results

Calculations of this paper have been done with different sets of boxes with different sizes with specified number. The number and types of boxes in experiments are selected in a way that the total volume of the box is almost equal to the volume of the container. According to the largest volume of box equal to 0.4 percent of the total volume of the container, the interior space of the container is divided into eight equal parts with a longitudinal and three transverse cut. Considered sets start with low diversity of 10 boxes and end with a great variety of 100 boxes. The higher the number of species is, the lower the average number of each type of boxes will be. Conversely the lower number of species is, the higher the average number of each type of boxes will be. The method was compared with the three methods presented in Table 3. The results are shown in Table 4. The mutation rate of algorithm MBRKGA is 0.02 and generations' number were considered as100. The algorithm stops if the search is completed. Computations have been performed through a processor Intel (R) Core (TM) i5-3210M CPU @ 2.50GHz, RAM 6G, windows 7 ultimate operating system and software versions MATLAB R2013a (8.1.0.604).

Table 3: Comparative Methods

Thapatsuwa et al.(2012)	Particle Swarm Optimization (PSO)
Thapatsuwa et al.(2012)	Artificial Immune System (AIS)
Goncalves &Resende.(2012)	A parallel multi population biased random key genetic algorithm(BRKGA)

Table 4: View of Container Volume Occupied by Different Algorithms for Different Collection Boxes.

PSO	AIS	BRKGA	MBRKGA	A variety of boxes
87.25	87.90	90.32	94.50	10
86.54	87.20	90.65	94.43	20
85.19	85.68	89.71	92.52	40
85.08	85.41	89.28	93.84	50
80.85	83.92	87.01	91.99	70
75.37	79.70	85.53	90.63	100

5. Conclusion

This paper considers the container loading problem by using proposed MBRKGA algorithm. The proposed method was compared with Particle swarm algorithm (PSO), BRKGA algorithms and algorithms immune systems (AIS) for solving the container loading problem. The results show that the method presented good results compared to other methods. Since the container space is divided into smaller parts, it can significantly increase the speed to achieve the results and the results will be far better than other similar methods. If different parts of the container are filled in parallel, this time will be much less.

Acknowledgements

The authors are grateful to the reviewers for their useful comments which helped to improve the quality of this manuscript.

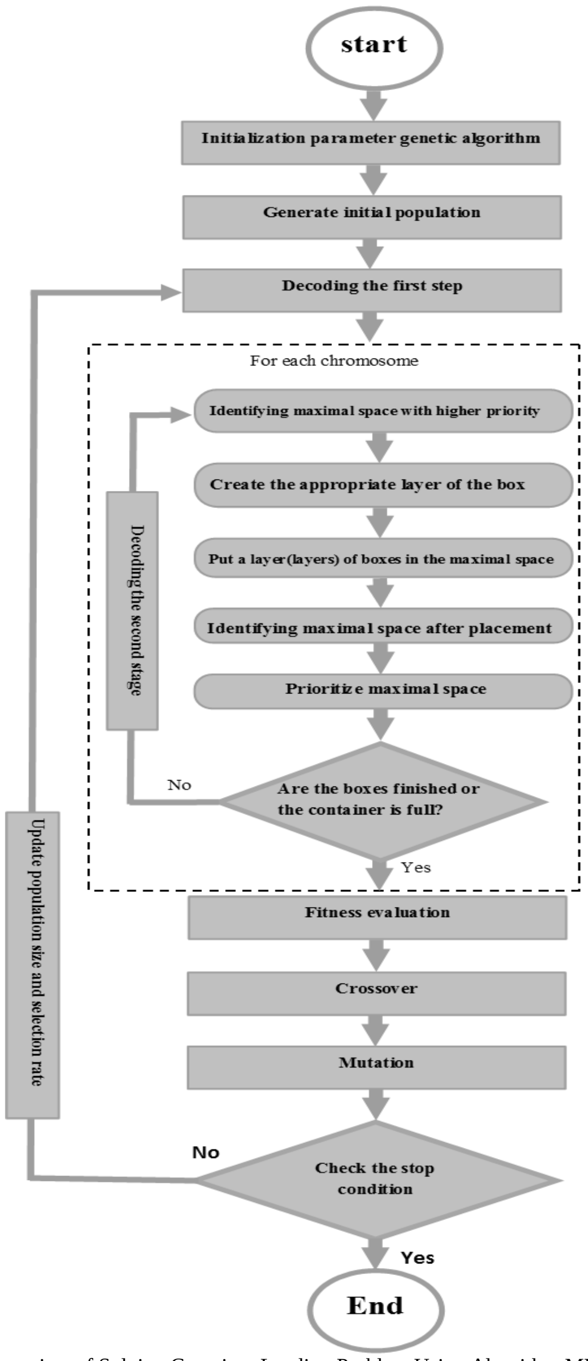

start
Initialization parameter genetic algorithm
Generate initial population
Decoding the first step
For each chromosome
Identifying maximal space with higher priority
Create the appropriate layer of the box
Put a layer(layers) of boxes in the maximal space
Identifying maximal space after placement
Prioritize maximal space
Decoding the second stage
No
Are the boxes finished or the container is full?
Yes
Fitness evaluation
Crossover
Mutation
Update population size and selection rate
No
Check the stop condition
Yes
End

Fig. 6: Overview of Solving Container Loading Problem Using Algorithm MBRKGA.

References

[1] Gonçalves, J.F., Resende, M.G.C., "A parallel multi-population biased random-key genetic algorithm for a container loading problem", Computers and Operations Research, Vol.39, No.2, (2012), pp.179–190, available online: http://dx.doi.org/10.1016/j.cor.2011.03.009.

[2] Gonçalves, J.F., Resende, M.G.C., "A biased random key genetic algorithm for 2D and 3D bin packing problems", Int. J. Production Economics, Vol.145, No.2, (2013), pp.500-510, available online: http://dx.doi.org/10.1016/j.ijpe.2013.04.019.

[3] W¨ascher G., Haussner, H., Schumann, H., "An improved typology of cutting and packing problems", European Journal of Operational Research, Vo1.83, (2007), pp. 1109–30, available online: http://dx.doi.org/10.1016/j.ejor.2005.12.047.

[4] Scheithauer, G., "Algorithm for the container loading problem", Operational Research Proceedings, (1992), pp. 445–452, available online: http://link.springer.com/chapter/10.1007%2F978-3-642-46773-8_112.

[5] Pisinger, D., "Heuristics for the container loading problem", European Journal of Operational Research. Vol. 141, (2002), pp. 143–53, available online: http://dx.doi.org/10.1016/S0377-2217(02)00132-7.

[6] Martello, S., Pisinger, D.,Vigo, D., "The three dimensional bin packing problem", Operations Research,Vol.48, No.2, (2000), pp. 256–67, available online: http://pubsonline.informs.org/doi/abs/10.1287/opre.48.2.256.12386. http://dx.doi.org/10.1287/opre.48.2.256.12386.

[7] George, AJ, Robinson, DF, "A heuristic for packing boxes into a container, Computers and Operations Research", Vol. 7, No.3, (1980), pp. 147–56, available online: http://dx.doi.org/10.1016/0305-0548(80)90001-5.

[8] H. Dyckhoff., "A typology of cutting and packing problems", European Journal of Operational Research,Vol. 44, No.2, (1990), pp.145–159, available online: http://dx.doi.org/10.1016/0377-2217(90)90350-K.

[9] Peeraya Thapatsuwan., Pupong Pongcharoen., Chris Hicks., Warattapop Chainate.," Development of a stochastic optimisation tool for solving the multiple container packing problems", Int. J. Production Economics, Vol.140, No.2, (2012), pp.737–748, available online: http://dx.doi.org/10.1016/j.ijpe.2011.05.012.

[10] Lai, K., Chan, J., "Developing a simulated annealing algorithm for the cutting stock problem", Computers and Industrial Engineering, Vol.32, No.1, (1997), pp.115–127, available online: http://dx.doi.org/10.1016/S0360-8352(96)00205-7.

[11] Davies AP., Bischoff EE., "Weight distribution considerations in container loading", European Journal of Operational Research, Vol.114, No.3, (1999),pp. 509–527,available online: http://dx.doi.org/10.1016/S0377-2217(98)00139-8.

[12] Gendreau, M., Iori, M., Laporte, G., Martello, S., "A tabu search algorithm for a routing and container loading problem", Transportation Science, Vol.40, (2006), pp. 342–350, available online: http://pubsonline.informs.org/doi/abs/10.1287/trsc.1050.0145. http://dx.doi.org/10.1287/trsc.1050.0145.

[13] Eley M., "Solving container loading problems by block arrangement", European Journal of Operational Research, Vol.141, No.2, (2002), pp.393–409, available online: http://dx.doi.org/10.1016/S0377-2217(02)00133-9.

[14] Bean JC., "Genetics and random keys for sequencing and optimization", ORSA Journal on Computing, Vol.6, No.2, (1994), pp.154–60, available online: http://dx.doi.org/10.1287/ijoc.6.2.154.

[15] Bortfeldt A., Gehring H., "A parallel genetic algorithm for solving the container loading problem", International Transactions in Operational Research, Vol.9, No.4, (2002), pp.497–511, available online: http://onlinelibrary.wiley.com/doi/10.1111/1475-3995.00369/abstract. http://dx.doi.org/10.1111/1475-3995.00369.

[16] Bischoff EE., Ratcliff MSW., "Issues in the development of approaches to container loading", Omega, International Journal of Management Science,Vol.23, No.3, (1995), pp.377–90, available online: http://dx.doi.org/10.1016/0305-0483(95)00015-G.

[17] Goldberg. D.,Holland, J., "Genetic Algorithms and Machine Learnin", Machine Learning, Vol.3, No.2, (1988), pp.95-99, available online: http://link.springer.com/article/10.1023%2FA%3A1022602019183?LI=true. http://dx.doi.org/10.1023/A:1022602019183.