OPTIMAL ARRANGEMENT OF RATION ITEMS INTO CONTAINER USING MODIFIED FOREST OPTIMIZATION ALGORITHM

Sathish Kumar Ravichandran
Assistant Professor, School of Engineering and Technology,
Department of Computer Science and Engineering, Christ University, Bengaluru, Karnataka
cbesathish@hotmail.com

Archana Sasi *
Assistant Professor, School of Engineering, Department of Computer Science and Engineering
Presidency University, Bengaluru, Karnataka
archana.sasi2k8@gmail.com

Abstract - Planning a shrewd framework for loading the ration goods into the container is one of the significant objectives in the mission of smart city advancement in India. This optimal container loading system is designed for the arrangement of ration goods into the container using Modified Forest Optimization algorithm for safe and secure delivery. The effectiveness of the proposed approach has been demonstrated using BR datasets and compared with different optimization algorithms. From the experiment, it is observed that the proposed Modified Forest Optimization algorithm is implemented in java meets the objective of loading the ration items into the container in an optimal fashion. Further, it is observed that the order of arrangement predicted by the proposed algorithm is found to be optimal than other competitive optimization algorithms..

***Keywords*:** Shrewd framework, Fair Price Shop, Smart City, Modified Forest Optimization algorithm, Container loading.

1. Introduction

Distributing food grains is shared between the central and state governments of India. The central government, specifically Food Corporation of India (FCI), is responsible for the interstate transport of food grains from procuring to consuming states, as well as delivering grains to the state godowns. Once FCI transports grains reach the state depots, distribution of food grains to the end consumers is the responsibility of state governments.

Traditional Public Distribution System (PDS) carries ration materials from warehouse to ration shops mostly using lorries that have the chances of dropping the goods falling down as the vehicle moves due to improper arrangements and thefts of ration materials. Further interruptions of natural calamities like rain, wind make the traditional PDS as an unsafe and unsecured transportation. The improper arrangement of ration goods in a lorry is shown in Figure 1.

Arranging the goods into the goods carrying vehicle is one of the major challenging tasks of PDS in India. This work focuses on providing a solution to such challenging task by using containers for the transportation of goods. The container is a kind of lorry like vehicle but it has a closed metal coverage that makes the goods free from the contamination of rain and other unsafe factors. Also, it comes in varying sizes, that make us transport more items in a single transmit for more ration shops so that count of lorries can be reduced. This container mode of transportation will reduce the manpower and fuel needed for transportation and will provide safer coverage for items than lorries. Large numbers of research papers are available in the literature that deals with the container loading using different optimization algorithm.

Xueping Li *et al.* (2015) proposed a Differential Evolution (DE) algorithm hybridized with a novel pressing heuristic technique, the best-coordinate first, which produced a minimized pressing arrangement in view of a given box pressing succession and a compartment stacking group. The adequacy of the proposed algorithm is assessed on an arrangement of mechanical occasions with haphazardly produced examples. The outcomes demonstrate that the proposed algorithm outflanks existing arrangement approaches regarding arrangement quality.

Fig. 1. Improper arrangements of ration goods in lorries

Tansel Dokeroglu *et al.* (2014) proposed an arrangement of hearty and adaptable crossover parallel algorithms that exploited parallel algorithm procedures, transformative gathering hereditary metaheuristics, for huge scale one-dimensional Bin Pressing Problem (BPP) occurrences. An aggregate number of 1318 benchmark issues is inspected with the proposed algorithms and it is demonstrated that ideal answers for 88.5% of these examples can be acquired with functional streamlining times while taking care of whatever the remains of the issues with one additional receptacle. At the point, when the outcomes are contrasted the current best in class heuristics in the parallel cross breed gathering hereditary algorithms can be considered as outstanding amongst other one-dimensional BPP algorithms as far as algorithm time and arrangement quality is considered.

Kyungdaw Kang *et al.* (2012) created a packing system to limit the number of cuboid spaces produced amid the packing procedure by coordinating the container height, length and width with the measurements of the pressing space. A half-breed hereditary algorithm was utilized to tackle the three-dimensional container packing issue with this developed system. The outcomes under the proposed algorithm were contrasted with different arrangements of computational outcomes found in scholastic papers identified with this subject.

D.S.LiuK.C.Tan *et al.* (2008) built up a Multi-Objective two-dimensional scientific model for Bin Packing Problem (MOBPP-2D). A multiobjective transformative molecule swarm improvement algorithm (MOEPSO) is proposed. Without the need of joining the two destinations into a composite scalar weighing capacity, MOEPSO consolidates the idea of Pareto's optimality to advance a group of arrangements along with the exchange of surface. Broad numerical examinations are performed on different test cases, and their exhibitions are contrasted both quantitatively, and factually and other improvement strategies to show the adequacy and proficiency of MOEPSO in taking care of the multiobjective receptacle bin packing issue.

Teodor Gabriel Crainic *et al.* (2016) built up a displaying system by presenting a class of container packing issues to be specific with the Stochastic Variable Cost and Size Bin Packing Problem. The subsequent stochastic variable cost detail with responding allocations to main stage of the strategic scope quantification choices of choosing containers, while the second stage models result acclimations to the arrangement, securing additional receptacles and packing the things into the container. Broad computational outcomes for a vast arrangement of cases that bolster the claim of legitimacy for the model, proficiency for the arrangement strategy proposed, and quality and heartiness of the arrangements are obtained.

Alessio Trivella *et al.* (2016) developed a multi-dimensional form with the useful augmentation for load balancing, i.e. to discover the packing require the base number of containers while guaranteeing the normal focus of mass of the stacked receptacles fall as close as conceivable to a perfect point. Giving trouble for the standard solvers to bargain even with little size examples, a multi-level nearby pursuit heuristic is displayed. The algorithm exploits the Fekete– Schepers portrayal of possible packing regarding specific classes of interim charts and iteratively enhances the heap adjustment of a container packing arrangement utilizing distinctive pursuit levels. The main level investigates the space of transitive introductions of the supplement diagrams related to pressing, the second adjusts the structure itself of the interim charts, the third trade things between container repacking n-tuples of pitifully adjusted receptacles. Computational examinations indicate exceptionally encouraging outcomes on an arrangement of 3D container packing problems.

Antonio G.Ramos *et al.* (2018) proposed a new load balance methodology for container loading problem in road transportation. In this work the author considers load balancing as a hard requirement and receives vehicle particular outlines that characterize the achievability space for the area of the focal point of gravity of the load, as indicated by the vehicle with particular specialized qualities, satisfying and agreeing to genuine directions along these lines and enactment. Multi-population biased random-key genetic algorithm is developed for static dependability and load balancing. Broad computational tests are performed with various variations of the proposed approach. The computational outcomes demonstrate that it is conceivable to get steady and load-adjusted arrangements without trading off the execution as far as container volume use and show likewise the preferred standpoint in consolidating load balancing using this bin packing algorithm.

A. Galrao Ramos *et al.* (2016) proposed a novel container loading algorithm for maintaining the static stability with static mechanical equilibrium conditions that applied to rigid bodies. The algorithm namely multi-population biased random-key genetic algorithm is developed to manage free spaces, and a layer building procedure to fill the maximal-spaces. Static stability condition is embedded in the placement procedure of the proposed algorithm using the evaluation function. The new algorithm is extensively tested on well-known literature benchmark instances using three variants: no stability constraint, the classical full base support constraint and with the new static stability constraint. A comparison is then made with the state-of-the-art algorithms for the container loading problem. The computational examinations demonstrate that by utilizing the new dependability foundation it is constantly conceivable to accomplish a higher level of container usage than with the established full base support imperative, for all classes of issues for ensuring static steadiness.

I. Araya *et al.* (2014) developed a container loading algorithm for three-dimensional packing problems with the goal to expand the space usage of the container. A new constructive approach namely beam search strategy is used as a variant of the branch-and-bound search that expands the most promising nodes at each level of the search tree. The approach is compared with other algorithms using 16 well-known sets of benchmark instances. Results demonstrate that the new approach outflanks all the others in each arrangement of examples.

Turkay Dereli *et al.* (2011) built a heuristic filling methodology with a new population-based optimization algorithm namely Bees Algorithm (BA). The proposed BA algorithm is intended to work with discrete factors. For this reason, distinctive operators (to achieve neighbourhood solutions) are utilized. The proposed hybrid BA algorithm is tried with two well-known sets of test problems available and implications of the results are also discussed.

Jose Fernando Goncalves *et al.* (2012) proposed a multi-population Biased Random-Key Genetic Algorithm (BRKGA) for a single container loading problem. The proposed algorithm hybridizes a novel position system with a multi-population genetic algorithm based on random keys. The BRKGA is used to evolve the order in which the box types are loaded into the container and the corresponding type of layer used in the placement procedure. A heuristic is utilized to decide the maximal space where each container is set. The approach is broadly tried on the total arrangement of test issue cases of Bischoff & Ratcliff and Davies & Bischoff and is contrasted with 13 different methodologies. The test set comprises of 1500 examples from the weakly heterogeneous payload. The computational trials show that it is not only just the approach performs extremely well in a wide range of occurrence classes yet in addition, it acquires the best general outcomes with contrasted and different methodologies.

Andreas Bortfeldt *et al.* (2001) designed a hybrid genetic algorithm for the single container loading problem with boxes of various sizes. Created stowage designs incorporate a few vertical layers with a few boxes. To produce offspring, specific genetic operators are used based on an integrated greedy heuristic. The procedure that considers a few functional requirements and test computations including methodology from different authors for the execution of the GA is discussed.

Chan Hou Che *et al.* (2012) designed Multiple Container Loading algorithms for Cost Minimization Problem (MCLCMP) with the goal to load products of various types into containers with different sizes in order to limit the aggregate cost. MCLCMP is changed over into a broadened set cover issue that defines using linear integer programming and solves it with a heuristic to generate columns. Examinations on standard container packing occasions demonstrate that the proposed approach is better than earlier methodologies.

A Bortfeldt *et al.* (2003) proposed parallel tabu search algorithm for the container loading problem. The parallel search is done by distinctively designed occurrences of a tabu search algorithm, which is collaborated by the trading of (best) arrangements toward the finish of characterized seek stages. The parallel inquiry forms are executed with a comparing number of LAN workstations. The productivity of the parallel tabu search algorithm is shown by a broad relative test including surely understood reference issues.

Chen-Fu Chien *et al.* (2009) built up a proficient computational methodology including three-dimensional cutting for deciding close ideal container loading examples to limit the misuse of holder space. The author utilized numerical cases from an engine organization that were imported scratch segments of Japan, produces parts in Taiwan and collects autos in China to evaluate its legitimacy and talked about the adequacy of the proposed arrangement.

Yao-Huei Huang et al. (2016) examined a three-dimensional single holder stacking issue that is expected to pack a given arrangement of unequal-measure rectangular boxes into a solitary container with the end goal that the length of the involved space in the container is limited. Spurred by the handy coordination occurrences the issue under examination is detailed as a zero-one blended whole number direct programming model. Because of the NP-hardness of the considered issue, a basic powerful loading position heuristic is proposed for illuminating substantial size examples. The exploratory outcomes exhibit that the created heuristic is fit for illuminating the cases with excess of two hundred boxes and is more productive than the cutting edge blended whole number direct program and existing heuristic strategies.

Christos D. Tarantilis *et al.* (2009) addressed a practical variant of the capacitated Vehicle Routing Problem (VRP) called the capacitated vehicle routing problem with 3-D loading constraints (3L-CVRP). This issue considers clients request to be shaped by 3-D rectangular things. Extra loading limitations frequently experience consideration of all things and utilization of transportation coordination's are forced on the analyzed issue for display the arrangements. Both issue renditions are fathomed by a mixture metaheuristic system that joins the procedures of Tabu Search (TS) and Guided Local Search (GLS). The loading qualities are handled by utilizing a gathering of pressing heuristics. The proposed algorithm's robustness is tested for both problem versions, solving benchmark instances derived from the literature and new benchmark problems with diverse features in terms of the customer set size and transported-item dimensions. It delivers fine outcomes, enhancing the vast majority of the best arrangements.

Leo Ho Wai Yeung et *al.* (2005) proposed a hybrid genetic approach for container loading in order to meet the up-to-date customer requirements in the logistic industry. Container loading is usually modelled as a three-dimensional packing problem which is known to be NP-complete. By combining an order-based genetic algorithm with a novel heuristic placement method, the complicated and highly constrained packing problem is transformed into a simple permutation problem with a much smaller searching domain. It is guaranteed that every box is stable after placement inside the container. From the simulation results, it is demonstrated that optimal results can be duly obtained reasonably in a short period of time.

Arranging the goods into the goods carrying vehicle is one of the major challenging tasks of Public Distribution System in India. This work focus on providing a solution to such challenge task by using containers for the transportation of goods. The container is a kind of lorry like vehicle but it has closed metal coverage that makes the goods free from contamination of rain and others. Also, it comes in varying size, that makes us to transport more items in a single transmit for more ration shops so that count of lorries can be reduced. This container mode of transportation will reduce the manpower, fuel that needed for transportation and safer coverage for items than lorries.

The rest of the paper is organized as follows. In section 2, Description of container loading problem is discussed. Overview of forest optimization algorithm is discussed in section 3. In section 4, Modified forest optimization algorithm for container loading is discussed. Pseudo code for Modified forest optimization algorithm for container loading is discussed in section 5. Simulation result is given in section 6. Finally concluding remarks with the scope for the future extension are discussed in section 7.

2. Container Loading Problem

Container loading problem can be construed as a geometric assignment problem, in which three dimension items (cargo) have to be packed into three dimension rectangular objects (containers) based on geometric feasibility conditions listed below.

- The boxes must not overlap each other
- All boxes must lie entirely within the container
- Each box can only be loaded through the container entrance
- Total weight loaded into the container should not exceed its maximum permissible weight

In general, the container loading problem may be stated as follows. Given a container *l* of dimensions L, W, H and a set of boxes C= {C_1, C_2, C_3…….C_N} the goal of the problem is to place boxes into the container with the objective of achieving a maximum utilization of the space of the container. Boxes do not overlap with one another and align completely inside the container with their faces parallel to the container faces. Formally, the objective may be stated as follows

$$\max \sum_{i=1}^{N} P_i * v_{(ci)} \tag{1}$$

Where $v_{(ci)}$ corresponds to the volume of the box c $_i$ and p $_i$ is a boolean variable indicating if the box *ci* is placed into the container.

The objective of this work is to arrange the ration items into a fixed size container in an optimal way so that the space of the container is effectively applied. The way of framing this problem attempts to obtain the optimized solution in the form of giving the best order of arrangements and achieving the maximum volume of containers.

3. Overview of Forest Optimization Algorithm

In this section, the steps involved in the execution of Forest Optimization Algorithm (FOA) are discussed.

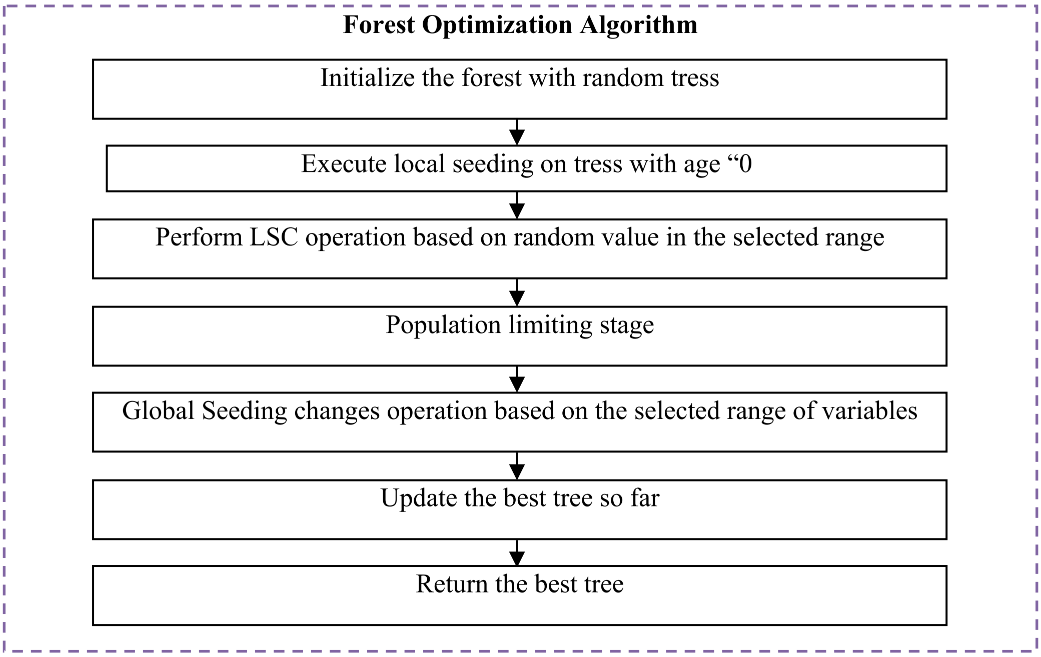

Forest Optimization Algorithm
Initialize the forest with random tress
Execute local seeding on tress with age "0
Perform LSC operation based on random value in the selected range
Population limiting stage
Global Seeding changes operation based on the selected range of variables
Update the best tree so far
Return the best tree

Fig 2. Schematic architecture of FOA

The schematic architecture of FOA is shown in Figure 2. This FOA algorithm is inspired by the growing and seeding procedure of the trees in the forest. This FOA involves three main stages namely 1.Local seeding of trees, 2. Population limiting, 3.Global seeding of trees. Like other evolutionary algorithm, FOA starts with the initial population of trees so that each tree represents a potential solution to the problem. Age is the numerical variable that is attached to each tree so that this variable defines the age of the particular tree and in the initial stage, age of the tree is set to 0. After the initialization of the trees, the local seeding operator will generate new trees from the tree with the age=0. The age of the tree is increased by 1 than other newly generated trees.

In population limiting stage trees that exceed the area limit in the forest are collected to form the candidate population for global seeding stage. In global seeding stage, a certain percentage of the candidate population is selected. From the selected population, it adds a new potential solution to get local optimum. Now the trees in the forest are ranked according to their fitness values and the tree with the biggest fitness value is selected as the best tree and finally, the age of the best tree is set to 0. These stages of operation will continue until the terminal criterion is met.

4. Modified Forest Optimization Algorithm for Container Loading

In this section, the optimal arrangement of goods into the container using MFOA is elaborately discussed below. The schematic architecture of the proposed work is shown in Figure 3.

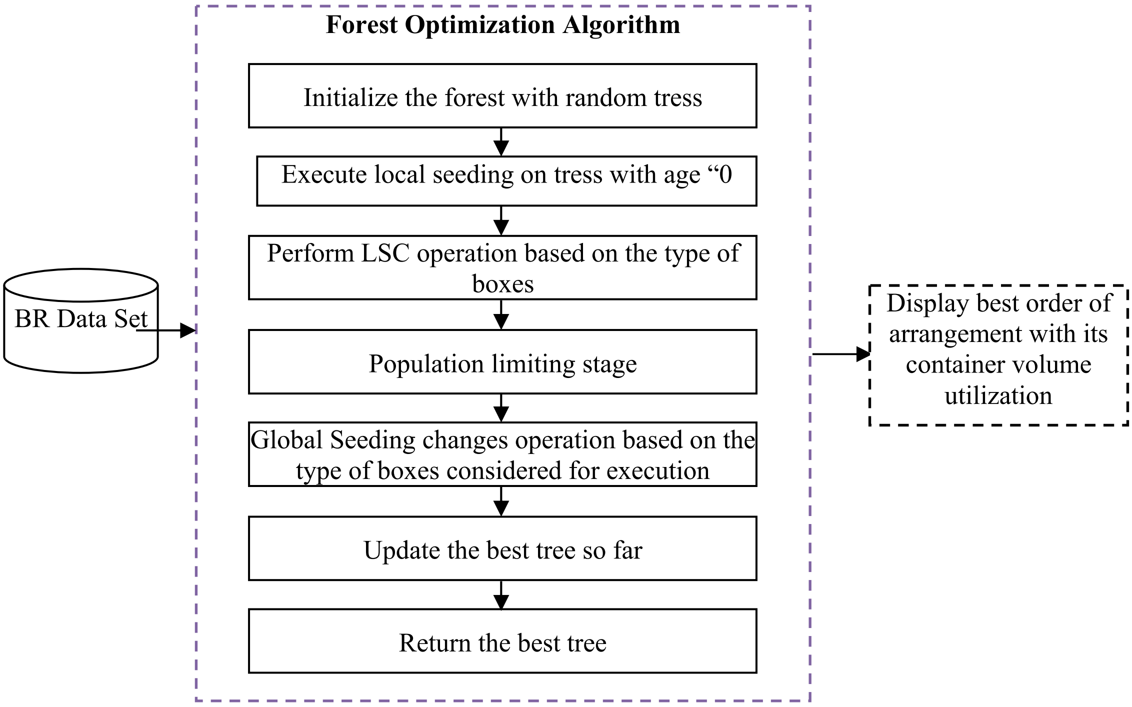

Forest Optimization Algorithm
Initialize the forest with random tress
Execute local seeding on tress with age "0
Perform LSC operation based on the type of boxes
Population limiting stage
Global Seeding changes operation based on the type of boxes considered for execution
Update the best tree so far
Return the best tree
BR Data Set
Display best order of arrangement with its container volume utilization

Fig 3. Schematic architecture of the proposed work

This MFOA is a new evolutionary algorithm suitable for continuous nonlinear optimization problems. It is inspired by the growing and seeding procedure of the trees in the forest. This MFOA involves three main stages namely 1.Local seeding of trees 2, Population limiting, 3.Global seeding of trees. Like another evolutionary algorithm, MFOA starts with an initial population of trees, so that each tree represents a potential solution to the problem. Age is the numerical variable that is attached to each tree so that this variable defines the age of the particular tree and in the initial stage age of the tree is set to 0.

After the initialization of the trees, the local seeding operator will generate the new trees from the tree with the age=0. The age of the tree is increased by 1 except newly generated trees. In population limiting stage, trees that exceed the area limit in the forest are collected and form candidate population for global seeding stage. In global seeding stage, a certain percentage of the candidate population is selected. From the selected population, it adds a new potential solution to get the local optimum. Now the trees in the forest are ranked according to their fitness values and the tree with the biggest fitness value is selected as the best tree and finally, the age of the best tree is set to 0. These stages of operation will continue until the termination criterion is met. The overall process flow for executing MFOA is explained as flowchart in Figure 4.

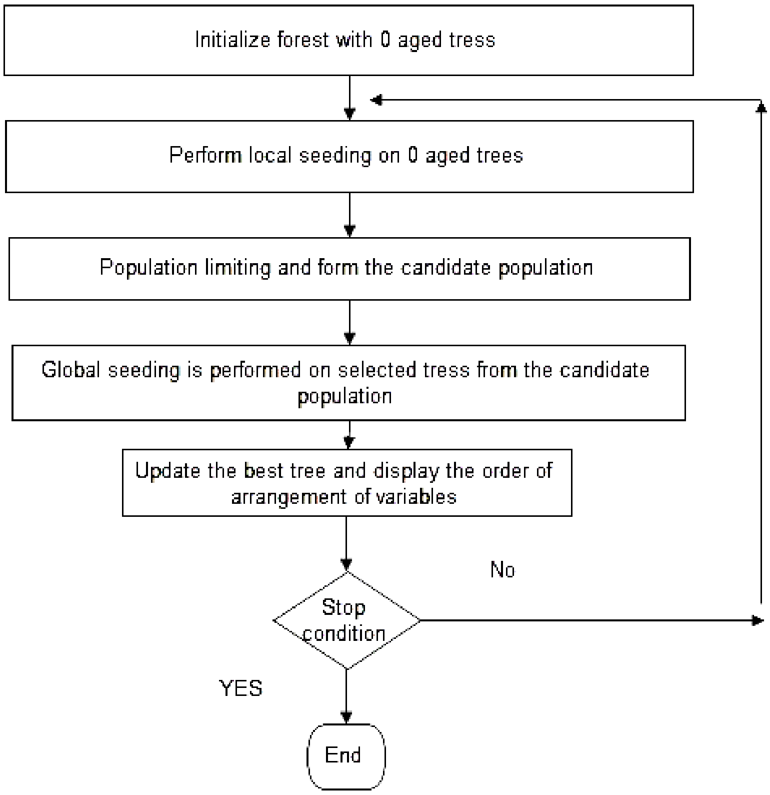

Initialize forest with 0 aged tress
Perform local seeding on 0 aged trees
Population limiting and form the candidate population
Global seeding is performed on selected tress from the candidate population
Update the best tree and display the order of arrangement of variables
Stop condition
No
YES
End

Fig. 4. Flowchart for loading the goods into the container using MFOA

The above Figure depicts the sequence flow of arranging the fair price shop goods into the container using MFOA. The detailed explanation of the proposed algorithm is discussed under subsections below.

4.1. *Initialize tress*

In MFOA the potential solution of each problem has been considered as a tree. Each tree shows the values of variables. In addition to the variables, each tree has in a related to the *"Age"* of that tree. After each local seeding stage, the age of trees, except new generated trees in local seeding stage, increases by '1'.The process of increasing the age of trees is used by a controlling mechanism for limiting the trees in the forest. Figure. 4.5 shows a solution representation of a N_{var} dimensional problem, where v_i are the values of the variables from the type of boxes and the *"Age"* part shows age of the related tree.

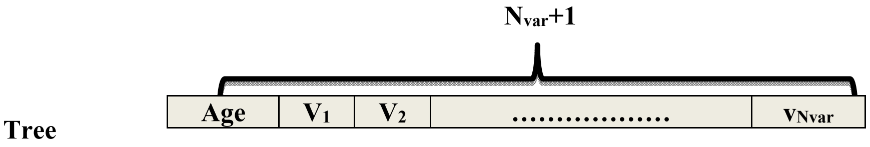


Fig. 5. Solution representation of MFOA.

A tree can also be considered as an array of length 1 x (N_{var}+ 1) as in Eq. (1), where N_{var} is the dimension of the problem and "Age" represents the age of the related tree.

$$\textbf{Tree} = [Age, v_1, v_2 ... v_{Nvar}] \qquad (2)$$

The maximum age of a tree is a predefined parameter and is named as *"lifetime"*. The *"lifetime"* parameter should be determined at the start of the algorithm. When a tree's "Age" reaches to the "lifetime" parameter, that tree is omitted from the forest and is added to the candidate population. If we choose a big number for this parameter, for each iteration of the algorithm the age of trees is just increased and the forest will be full of old trees, which do not take part in the local seeding stage or if we choose a very small value for this parameter, the trees will get old very soon and they will be omitted at the beginning of the competition. Therefore, this parameter should provide a good chance of local search.

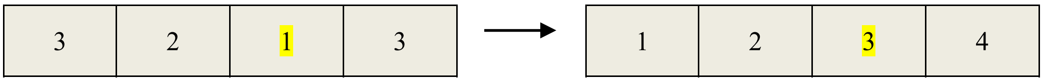
4.2. *Local seeding of the trees*

In nature when seeding procedure begins, some seeds fall just near the trees and after some time they turn into young trees. This local seeding of trees attempts to simulate this procedure of nature. This operator is performed on the trees with age 0 and adds neighbors of each tree to the forest. At this stage, after the local seeding executed on the trees at age 0, the age of all trees, except new generated trees is increased by 1.

Increasing the age of trees acts as a controlling mechanism on the number of trees of the forest. If a tree is promising, the procedure of the algorithm resets the age of that tree to ‘0’ the result, will be possible to add neighbors of the good tree to the forest through performing local seeding stage, otherwise non-promising trees get old with each iteration of the algorithm and finally die after some iterations.

The number of seeds that fall on the land near the trees turning into trees as neighbors are considered as a parameter of this algorithm that is named as “Local Seeding Changes” or “LSC”. At the first iteration of the algorithm, the trees with age 0 perform local seeding operation and new trees are added to the forest. At the next iteration, the number of added trees by local seeding operator decreases because there are trees with the age more than 0, which do not take part in local seeding stage.

In order to perform this LSC operator, a variable is chosen randomly from the selected tree. Then its value is changed with respect to the type of boxes that we used in our execution. Here we use three types of boxes namely 1, 2, and 3 and this procedure is repeated for LSC times for each tree with age 0.



3	2	1	3	→	1	2	3	4

Fig 6. Numerical example for local seeding operator

A numerical example for local seeding operator is shown in Figure.6. The value of LSC is 1 in Figure. 6 variable number 1 is selected and it is changed into 3 with a randomly generated value from the type of boxes used for our execution. Now the new tree with age 0 will be added to the forest. Local seeding operator adds many trees to the forest, so there must be a limitation on the number of trees. This control is done in the next stage of the proposed algorithm.

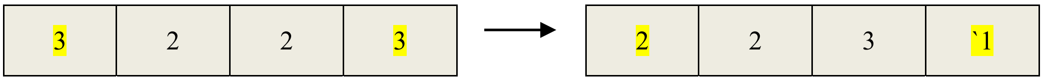
4.3. *Population limiting*

The number of trees in the forest must be limited to prevent infinite expansion of the forest. There are 2 parameters which limit the population of trees namely “Lifetime” and “area limit”. In the first stage, the trees whose age exceeds the “lifetime” parameter are removed from the forest and they are moved to the candidate population separately. In the second stage limitation parameter (i.e.) “area limit” is used. After ranking the trees according to their fitness values if the number of the trees is greater than the limitation of the forest, then the extra trees are removed from the forest and are added to the candidate population. So, after performing these two operations, the number of trees in the forest will be equal to the number of initial trees. After population limiting of the forest, global seeding stage is performed on some percentage of the candidate population.

4.4. *Global seeding of the trees*

The seeds of the trees are dispersed in the entire forest and as a result, the habitat of the trees becomes wider. Also, other natural processes like wind and flow of water support to distribute the seeds in the entire forest. Global seeding stage attempts to simulate the distribution of the seeds of the trees in the forest. Global seeding operator is performed on a predefined percentage of the candidate population using “transfer rate” parameter.

Global seeding operator works as follows: At first, the trees from the candidate population are selected. Then, some variables for each tree are selected randomly. The value of each selected variable is exchanged with another randomly generated value from the type of boxes. As a result, a tree with age 0 is added to the forest. This operator performs a global search in the search space. The number of variables, whose values will be changed, is another parameter of the algorithm and is named as “Global Seeding Changes” or “GSC”.



3	2	2	3	→	2	2	3	`1

Fig 7. Numerical example for global seeding operator

Figure 7 is an example of performing global seeding operator for one tree in the continuous space. The value of “GSC” parameter is considered as 2. Two variables are selected randomly and their values are exchanged with two other randomly generated values from the type of boxes used for our execution.

4.5. *Updating the best tree*

After sorting out the trees according to their fitness values, the tree with the highest fitness value is selected as the best one and the arrangement of the variable of the best tree is also displayed.

4.6. *Stop condition*

Like other evolutionary algorithms, three stop conditions are considered 1- Predefined number of iterations 2- Observance of no change in the fitness value of the best tree for several iterations 3- Reaching to the specified level of accuracy. The 3 main stages of proposed MFOA in container loading is discussed in the next chapter.

5. Pseudo Code of MFOA for Container Loading

In this section, pseudo code for the proposed MFOA for container loading is discussed.

```
Input: lifetime, LSC, GSC, transfer rate, area limit.
Output: The best tree with its order of arrangement of variables is displayed.
Step-1: Initialize the forest with random trees
    1.1 Each tree is represented as a dimensional vector X, then X= (age, v1, v2, v3…., vn) for D
dimensional problem.
    1.2 "Age" of each tree is initialized by "0".
Step-2 While stop condition is not satisfied do
    2.1: Perform local seeding on trees with age 0
•   For i=1 "LSC"
-   Randomly choose a variable from the selected tree.
-   Change the selected variable of the tree randomly from "B" used for execution.  Here B= {1, 2, 3}.
•   Increase the age of all the trees by 1 except newly generated trees.
        2.2 Population limiting
•   Remove the trees with the age greater than "lifetime" parameter and add them to the candidate population.
•   Sort out the trees according to their fitness value
•   Remove extra trees that exceed "area limit" parameter from the end of the forest and add them to the
    candidate population.
        2.3 Global Seeding
•   Choose "transfer rate" percent of the candidate population
•   For each selected tree
    - Choose "GSC" variable to the selected tree randomly
    - Change the value of each variable with another randomly generated variable from "B" and add a new tree
    to the forest with age=0.
        2.4 Updation of the best tree so far
                Sort trees according to their fitness value
                Reset the age of the best to 0
    Step 3: Return the best tree
```

6. Simulation Result

This section presents the details of the simulation carried out on BR datasets. Simulations are conducted to examine the percentage of container volume utilization by the proposed MFOA. The proposed MFOA approach is implemented in Java SE 6 and executed in a PC with Intel Core i3 processor, 2.4 GHz speed and 4 GB of Ram.

6.1. *Datasets for Container Loading Problem*

The datasets considered in the simulation are proposed by Bischoff & Ratcliff (1995) and Davies & Bischoff (1999). The data set is classified into 10 classes, with a total of 25 instances per class and designated as BR1 to BR10. The dataset details used for our simulation are listed in Table 1.

Table 1. Details of the Datasets

S.no	Dataset name	Number of instances	Type of Boxes
1	BR-1	25	3
2	BR-2	25	5
3	BR-3	25	8
4	BR-4	25	10
5	BR-5	25	12
6	BR-6	25	15
7	BR-7	25	20
8	BR-8	25	30
9	BR-9	25	40
10	BR-10	25	50

6.2. *Control Parameter Setting*

The MFOA was run with different number of trees and MFOA control parameters. Computational experiments allowed the selection of parameters using BR datasets with the following settings are presented in Table.2.

6.3. *Performance Computation using MOFA*

The arrangement of items and their container utilization are of two performance metrics studied to assess the performance of the proposed MFOA algorithm. Table 3 reports the value obtained for those metrics individually for each number of data from BR1 to BR10.

Table 2. Forest optimization algorithm parameter used in computational experiments

S.No	Parameter	Value
1	Initial population trees in the forest	30
2	Area limit	30
3	Local seeding changes	1
4	Global seeding changes	1
5	Transfer rate	10%
6	Lifetime	6

Table 3. Performance computation using MFOA algorithm

S.No	**Number of boxes**	**Container utilization (%)**
1	BR1	95.30
2	BR2	96.27
3	BR3	96.51
4	BR4	96.33
5	BR5	96.22
6	BR6	96.08
7	BR7	95.79
8	BR8	95.35
9	BR9	95.09
10	BR10	94.99

The above table shows the volume utilization of the container for different BR data sets. The fitness of the obtained solution is evaluated through the fitness function namely _f(x) = l*b*h of the box / l*b*h of the container *100_ → (1) From this table it is observed that the proposed MFOA algorithm obtain the best order of arrangement and their container utilization for the data set BR1 to BR10 For simplicity, the best order of arrangement using BR-1 data set is 3, 2, 2, 1, 2, 3, 2, 1, 3, 2, 1, 2, 3, 3, 3,2, 2, 3, 3, 2, 2, 1, 3, 2, 3, 2, 2, 3, 1, 1, 1, 1, 1, 2, 2, 3, 3, 3, 3, 2, 1, 2, 2, 1, 3, 3, 3, 1, 2, 1, 2, 1, 2, 2, 3, 3, 2, 3, 3, 1, 3, 2, 2, 1, 1, 3, 3, 2, 1, 3, 3, 2, 1, 3, 1, 3, 3, 2, 1, 2, 1, 3, 2, 2, 3, 3, 1, 1, 1, 3, 1, 1, 2, 2, 1, 2, 2, 1, 2, 1, 2, 2, 3, 1, 2, 3, 1, 3, 1, 3, 1, 2. The fitness calculation is done by using (1). The 1st type of box with length=108(m) width=76(m) and height=30(m), the 2nd type of box with length=110(m) width=43(m) and height=25(m), the 3rd type of box with length=92(m) width=81(m) and height=55(m) and the container with length=587(m), width=223(m) and height=220 (m).

The fitness calculation is done in 3 steps. In the first step, counting the number of occurrences of the boxes and multiplying with their corresponding length, width and height. In the second step, the solution obtained by step 1 is divided by the length, width, height of the container. In step 3, the solution obtained by step 2 is multiplied by 100 and finally, the volume utilization is achieved for each number of data from BR1 to BR10

6.4. *Performance Comparisons*

To compare the performances of the proposed MFOA, three different approaches are developed and the comparison is given in Table 4.

Table 4 Comparison of different optimization algorithms

		Container volume Utilization in %			
S.No	**Number of Datasets**	**GA**	**ABC**	**ACO**	**MFOA**
1	BR1	94.82	93.51	95.02	**95.30**
2	BR2	95.40	94.45	96.07	**96.27**
3	BR3	95.55	95.10	96.21	**96.51**
4	BR4	95.37	95.14	96.10	**96.33**
5	BR5	95.20	95.01	95.90	**96.22**
6	BR6	95.09	95.20	95.85	**96.08**
7	BR7	94.60	94.55	95.33	**95.79**
8	BR8	94.12	94.64	94.77	**95.35**
9	BR9	93.70	94.29	94.45	**95.09**
10	BR10	93.37	94.12	94.65	**94.99**

The first one is Genetic Algorithm (GA) that uses binary strings for representation of solution variables and genetic operators such as tournament selection, two-point crossover and bitwise mutation. The second one is Artificial Bee Colony (ABC) algorithm that uses the position of food source with colony size, limit and max cycle as the control parameters. The last one evolves the control parameter using standard Ant Colony optimization (ACO).

From this Table.4, it is found that the proposed MFOA has higher volume utilization than other approaches of all BR datasets. The objective of this work is to meetout and arrange the goods into the container in an optimized way by utilizing the maximum container volume for loading ration goods to distribute for various fair price shops by the developed MFOA algorithm.

6.5. *Convergence Analysis*

The convergence analysis of the proposed MFOA for container loading is tested by running the algorithm for several iterations and the result is displayed in Figure 8.

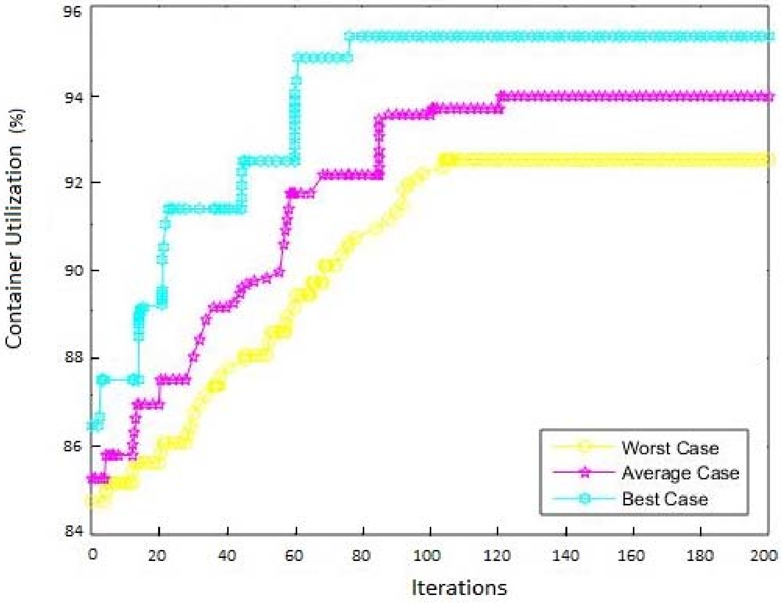


Fig. 8 Convergence Analysis of MFOA

The stability behavior of the proposed MFOA for container loading is tested by running the algorithm for 30 individual runs and the result are displayed in Figure 9.

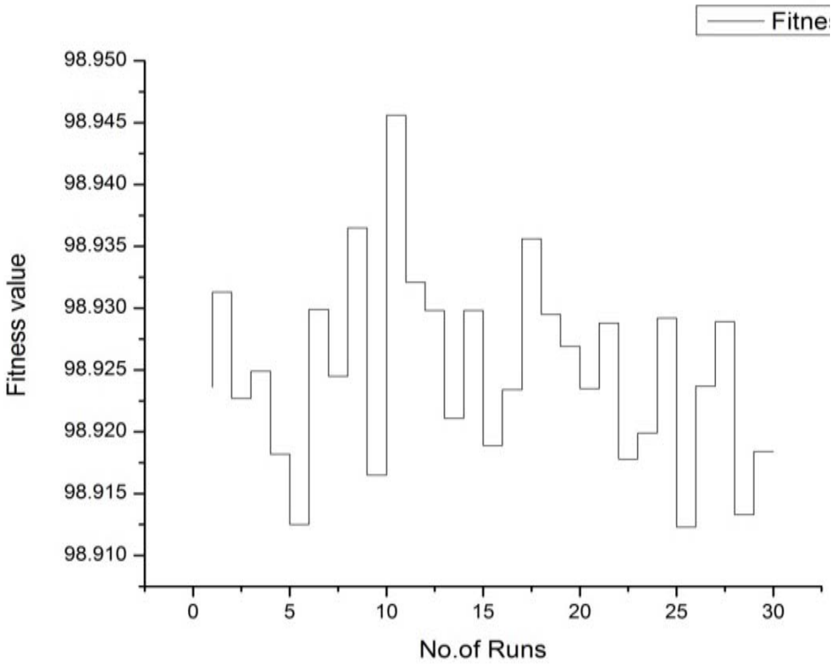


Fig. 9 Stability behaviour of the proposed MFOA

Figure 9 denotes the stability behaviour of the proposed MFOA for container loading using fitness function evaluation. The horizontal axis denotes the total number of runs, where as the vertical axis denotes the fitness value. From the experiment, it is observed that the proposed algorithm has obtained the stability in the successive runs from 98.91% to 98.95%on volume utilization for loading the goods into the container.

7. Conclusion

A novel Modified Forest Optimization algorithm is implemented for arranging ration goods into the container for safe and secure delivery of ration items to various fair price shops. The effectiveness of the proposed approach has been demonstrated using BR data sets and compared with a different optimization algorithm. This kind of computational arrangement of ration materials into the goods carrying container is more secure and useful for Indian Civil Supplies Corporation (ICSC) to carry the food materials to different fair price shops within India.

References

[1] Xueping Li, Kaike Zhang 2015, ‘A hybrid differential evolution algorithm for multiple container loading problem with heterogeneous containers’, Computers & Industrial Engineering, vol. 90, no. 7, pp. 305-313.

[2] Tansel Dokeroglu, Ahmet Cosar 2014, ‘Optimization of one dimensional Bin Packing Problem with island parallel grouping genetic algorithms’, Computers & Industrial Engineering, vol. 75, no. 3, pp. 176-186.

[3] Kyungdaw Kang, Ilkyeong Moon, HongfengWang 2012, ‘A hybrid genetic algorithm with a new packing strategy for the three dimensional bin packing problem’, Applied Mathematics and Computation, vol. 219, no. 3, pp. 1287-1299.

[4] D.S.Liu K.C.Tan, S.Y.Huang, C.K.Goh, W.K.Ho 2008, ‘On solving multiobjective bin packing problems using evolutionary particle swarm optimization’, European Journal of Operational Research, vol. 190, no. 2, pp. 357-382.

[5] Teodor Gabriel Crainic, Luca Gobbato, Guido Perboli, Walter Rei 2016, ‘Logistics capacity planning: A stochastic bin packing formulation and a progressive hedging metaheuristic’, European Journal of Operational Research, vol. 253, no. 2, pp. 404-417.

[6] Alessio Trivella, David Pisinger 2016, ‘The load balanced multidimensional bin packing problem’, Computers & Operations Research, vol. 74, no. 4, pp. 152-164.

[7] Antonio G.Ramos, Elsa Silva, Jose F. Oliveira 2018, ‘A new load balance methodology for container loading problem in road transportation’, European Journal of Operational Research, vol. 266, no. 3, pp. 1140-1152.

[8] Galrao Ramos, Jose F. Oliveira, Jose F. Goncalves, Manuel P. Lopes 2016, ‘A container loading algorithm with static mechanical equilibrium stability constraints’, Transportation Research Part B: Methodological, vol. 91, no. 3, pp. 565-581.

[9] I. Araya, M. C. Riff 2014, ‘A beam search approach to the container loading problem’, Computers & Operations Research, vol. 43, no. 3, pp. 100-107.

[10] Turkay Derelia, Gulesin Sena Das 2011, ‘A hybrid ‘bee(s) algorithm’ for solving container loading problems’, Applied Soft Computing, vol. 11, no. 2, pp. 2854-2862.

[11] Jose Fernando Goncalves, Mauricio G.C.Resende 2012, ‘A parallel multipopulation biased random key genetic algorithm for a container loading problem’, Computers & Operations Research, vol. 39, no. 2, pp. 179-190.

[12] Andreas Bortfeldt, Hermann Gehring 2001, ‘A hybrid genetic algorithm for the container loading problem’, European Journal of Operational Research, vol. 131, no. 1, pp. 143-161.

[13] Chan Hou Che, Weili Huang, Andrew Lim, Wenbin Zhu 2011, ‘ The multiple container loading cost minimization problem’, European Journal of Operational Research, vol. 214, no. 3, pp. 501-511.

[14] A Bortfeldt, H Gehring, D Mack, ‘A parallel tabu search algorithm for solving the container loading problem’, Parallel Computing, vol. 29, no. 5, pp. 641-662.

[15] ChenFu Chien, Chia YenLee, YiChao Huang, WenTingWu 2009, ‘An efficient computational procedure for determining the containerloading pattern’, Computers & Industrial Engineering, vol. 56, no. 3, pp. 965-978.

[16] YaoHuei Huang, F.J. Hwang, HaoChun Luc 2016, ‘An effective placement method for the single container loading problem’, Computers & Industrial Engineering, vol. 97, no. 3, pp. 212-221.

[17] Christos D. Tarantilis, Emmanouil E. Zachariadis, Chris T. Kiranoudis 2009, ‘A Hybrid Metaheuristic Algorithm for the Integrated Vehicle Routing and ThreeDimensional Container Loading Problem’, IEEE Transactions on Intelligent Transportation Systems, vol. 10, no. 2, pp. 255-271.