Effects of Memory and Genetic Operators on Artificial Bee Colony Algorithm for Three-Dimensional Bin Packing Problem

Tuğrul Bayraktar, Filiz Ersöz and Cemalettin Kubat

June 7, 2021

Effects of Memory and Genetic Operators on Artificial Bee Colony Algorithm for Three-Dimensional Bin Packing Problem

Tuğrul Bayraktar[1], Filiz Ersöz[1] and Cemalettin Kubat[2]

[1] Karabuk University, Department of Industrial Engineering, Turkey
tugrulbayraktar@karabuk.edu.tr, fersoz@karabuk.edu.tr

[2] Hasan Kalyoncu University, Department of Industrial Engineering, Turkey
kubat@sakarya.edu.tr

KEYWORDS - Artificial Bee Colony Algorithm, Tabu Search, Genetic Algorithm, Three-Dimensional Bin Packing Problem, Knapsack Problem

ABSTRACT

The Artificial Bee Colony (ABC) algorithm is widely used to achieve optimum solution in a short time in integer-based optimization problems. However, the complexity of integer-based problems such as Knapsack Problems (KP) requires robust algorithms to avoid excessive solution search time. ABC algorithm that provides both the exploitation and the exploration approach is used as an alternative approach for various KP problems in the literature. However, it is rarely used for the Three-Dimensional Bin Packing Problem (3DBPP) which is an important part of the transportation systems. In this study, the exploitation and exploration aspects of the ABC algorithm are improved by using memory mechanisms and genetic operators to develop two different hybrid ABC algorithms. The developed algorithms and the basic ABC algorithm are applied to a generated 3DBPP dataset to observe the effects of the memory mechanism and the genetic operators separately. The results show that the genetic operators are more effective than the memory mechanism to develop a hybrid ABC algorithm, for solving heterogeneous 3DBPPs.

1 INTRODUCTION

Containers are one of the basic elements of transportation networks. Commonly used containers to be filled by many goods for distribution to the same or different locations have a variety of dimensions. Allocating items into limited spaces, is a combinatorial optimization problem and bin packing problem (BPP) is a branch of knapsack problems (KP), where a set of items is loaded into multiple capacitated bins. If the sizes of items and bins differ in all three direction, this type of problem is called as the three-dimensional bin packing problem (3D-BPP) [1].

First part of 3D-BPP researches focused on to improve the mathematical modellings [2] for better item allocation approach, the second part of the researches proposed solution approaches [3] for variable bin packing conditions and constraints, and the last part of the researches has improved the item allocation orders with heuristic methods [4]–[6] to provide better solutions compared to the integer-based programming approaches.

Artificial Bee Colony (ABC) algorithm is a neuro-inspired meta-heuristic approach based on foraging behavior of the bee colonies and the effectiveness of the ABC algorithm has been proved by many one-dimensional KP researches [7], [8] beside the previous work [9]. However, the ABC algorithm has been scarcely used for 3D-BPP and this study aims to contribute to this branch of BPP, by using the ABC algorithm as a solution search approach.

Genetic Algorithm (GA) based on improving the randomly generated individuals of the population thorough iterations, and Tabu Search (TS) based on restricting the search moves to explore the best problem solution, are other heuristic methods commonly used for KP and can likewise be used for 3D-BPP. Gehring and Bortfeldt [10] proposed one of the first GAs for the Single Container Loading Problem (SCLP), a branch of the KP, to satisfy the loading constraints when allocating items to the container. Bortfeldt and Gehring [11] also proposed a hybrid GA to optimize the container loading plans by building layers of well-classified items. Wu et al. [12] used GA for Strip Packing Problem (SPP) to optimize the bin packing plan to determine the height of bins in use. Kang et al. [4] used GA to minimize the number of rectangular residual spaces in the bins to reduce the number of bins in use.

Bortfeldt and Gehring [13] also implemented TS for SCLP to minimize the volume of rectangular residual spaces in the container by selecting the best item in each allocation. Liu et al. [14] used a hybrid TS that selects items not only individually but also as groups in each allocation and provides alternatives for container loading solutions by allocating the assigned items vertically or horizontally. Mack et al. [15] used TS as a reinforcement to the hybrid algorithm to avoid abandoned solutions to be re-generated over a period of time. Zhu et al. [16] used TS to select one of item placement strategies (deep-bottom-left or maximum touching area) in each allocation step to obtain the optimum loading solution.

Genetic operators (mutation and crossover) in GA, and TS strategies has been used as reinforcement approaches to strengthen the exploration and the exploitation aspects of the ABC algorithm respectively. Ozturk et al. [8] and Panahi and Navimipour [17] used genetic operators to improve the exploration capability of the ABC algorithm by increasing the number of alternative solutions around existing ones in the population. The new alternative solutions were generated in both employed bee and onlooker bee phase, using the crossover operator between the current solution, two random neighbor solution, zero solution and best solution obtained. All the solutions obtained after crossover were mutated before choosing the best alternative as the new solution among the current and generated solutions. Chaurasia et al. [18] simplified the search process in the ABC algorithm by using 3-point insertion method to generate new solutions from the current one and one of its random neighbors. In this way, the number of function evaluations was reduced, compared to the approach in [8].

TS strategies avoid the repetitive search steps while generating new solutions by classifying steps as efficient or inefficient to save in different tabu lists. Chengli et al. [19] integrated a memory mechanism into the ABC algorithm to save successful parameter pairs for reuse in next iterations to increase the probability of escaping local optimum. In the previous study [9], a memory mechanism was integrated into the ABC algorithm to save inefficient solutions into the Short-Term Tabu List (STTL) to improve the exploitation ability in the neighborhood search phase.

This study aims to improve the ABC algorithm separately with genetic operators and a memory mechanism, and then observe the effects of the two reinforcement approaches on the ABC algorithm. The basic ABC algorithm, the memory integrated ABC (MIABC) algorithm and the genetic operator-based ABC (GABC) algorithm were separately applied to a generated 3D-BPP data set. The rest of the paper organized follows: the brief explanation of 3D-BPP and the way of its implementation in this study is provided in Section 2; the proposed solution approaches are explained in Section 3; results from applied approaches are represented in Section 4, and Section 5 concludes with the future work and inferences about research undertaken.

2 3D BIN PACKING PROBLEM

3D-BPP is a branch of KP based on placing the entire set of 3D elements into as few 3D bins as possible, so the aim is to maximize the average utilization ratio (UR) of the bin in use without item intersection and dimensional exceeding. The average utilization ratio is calculated as shown in Eq. (1) and Eq. (2), without considering any other variables but the volume of the assigned items:

$$maximize \ \frac{\sum_{b=1}^{C}\sum_{i=1}^{n} a_{ib}.v_{ib}}{\sum_{b=1}^{C} V_b} \tag{1}$$

$$subject\ to \ \sum_{i=1}^{n} a_{ib}.v_{ib} \ \leq V_b \tag{2}$$

The average UR in Eq. (1) equals the sum of the volumes of all items assigned to the bins divided by the total volume of the bins used where $a_{ib} \in [0,1]$ is an item assignment variable, v_{ib} is the volume of the assigned item $i = 1, \dots, n$, and V_b is the volume of bin $b = 1, \dots, C$. In this study, V_b is identical for all bins. The total volume of the assigned items in each bin cannot be larger than the capacity of the bin as shown in Eq. (2).

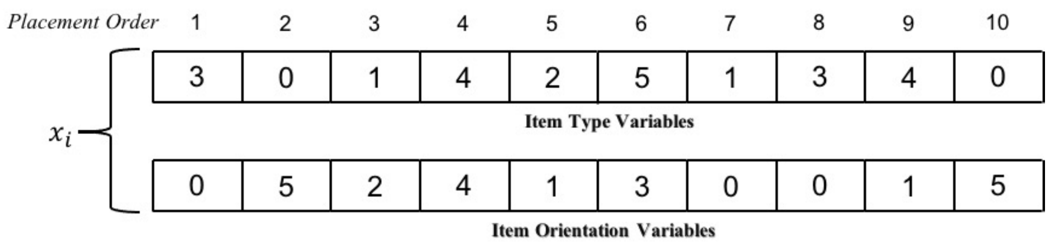

Figure 1: Bin Packing Sequence

In this study, the items are divided into five types for each problem set, and each item type can be placed in the bin in six orientation. The bin packing process begins by generating a packing sequence consisting of the item types $t = 0, \dots, 5$ in the first row and the item orientation $o = 0, \dots, 5$ in the second row in each column, as shown in Figure (1). Types and orientations are randomly placed in the packing sequence in pairs. Then, the items are placed into the first bin according to the placement order using the deep-bottom-left first (DBLF) item placement approach. Whole packing sequence is assigned to bins according to first-fit approach, that allocates the next item, starting with the first bin each time, whichever bin it can fit in first.

3 HEURISTIC APPROACHES FOR 3D BIN PACKING PROBLEM

3.1 Artificial Bee Colony Algorithm for 3D Bin Packing Problem

The ABC algorithm was developed by Karaboga and mimics the foraging behaviors of honeybees that are divided into three groups; employed bees, onlooker bees and scout bees. The ABC algorithm was originally designed for numerical problems [20] and eventually modified for integer-based problems [21] due to the easy applicability and the search simplicity.

The initial population in the basic ABC algorithm is generated from a randomly chosen solution using Eq. (3) for numerical problems, where $i = 1...SN$ refers to the i-th food source and SN refers to the total number of bees and food sources in the search area. $j = 1...D$ refers to the j-th dimension value of the i-th food source between an upper and a lower bound and D refers to the total number of parameters of the i-th food source to optimize. In this study, the initial population is generated from a random packing sequence by replacing one pair of type and orientation variables with a randomly selected pair, in each generation process.

$$x_{ij} = x_j^{min} + rand(0,1)\left(x_j^{max} - x_j^{min}\right) \tag{3}$$

In the employed bee phase, each employed bee visits only one solution to generate a new one using Eq. (4) for numerical problems, where v_{ij} is the new solution generated from the interaction between the same j elements of visited solution x_i and its neighbor solution x_k, and the difference between x_i and x_k is weighted by the ϕ_{ij}, which takes values between $[-1, 1]$. In this study, the new solution is generated according to binary optimization scheme using Eq. (5), where the $\oplus$ symbol is an xor operator [22] corresponding to the $(-)$ operator in Eq. (4), to measure the difference between x_i and x_k. If the fitness value f_{v_i} is better than the visited solution x_i's, v_i replaces it. Otherwise, the failure counter of the visited solution $failure_i$ is increased by one.

$$v_{ij} = x_{ij} + \phi_{ij}\left(x_{ij} - x_{kj}\right) \tag{4}$$

$$v_{ij} = x_{ij} \oplus \phi\left(x_{ij} \oplus x_{kj}\right) \tag{5}$$

In the onlooker bee phase, the bees in the hive evaluate the fitness values of the solutions calculated in Eq. (1) and choose one of them with the probability p_i calculated as in Eq. (6). If the onlooker bees improve the current food sources, they memorize the new food sources and forget the old one. Otherwise, $failure_i$ value is increased by one again.

$$p_i = f_i \Big/ \left(\sum_{i=1}^{SN} f_i\right) \tag{6}$$

If the maximum failure exceeds the failure limit L, the onlooker bee abandons the food source, except the food source with the best quality, and turns into the scout bee that explores new food sources randomly by using Eq. (3). In each of the iteration, only one onlooker bee is allowed to become a scout bee.

3.2 Memory Integrated Artificial Bee Colony Algorithm

The honeybees in ABC algorithm forget all the information about the improvement process, once they abandon the food source that reaches the maximum failure limit in the population. However, the information about a succeeded or failed move from x_i to x_j can be used by other honeybees to accelerate the search process. In this study, the search moves ($move_{ij}$) are represented as arrays in which the processed item with the related elements in x_i and x_j are saved, as shown in Fig. (2). In the $move_{ij}$ array, the first element indicates the packing order of the item and the placement information about the item in x_i and x_j is saved respectively in the rest of the array.

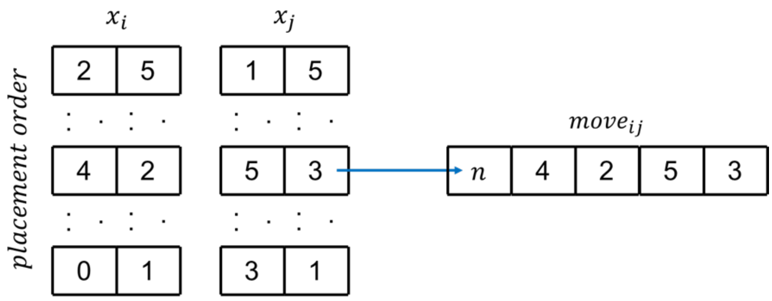

Figure 2: The Memory Mechanism for ABC Algorithm

If $move_{ij}$ improves the solution in the employed and the onlooker bee phase, it is saved in Intermediate-Term Tabu List (ITTL). Besides, $move_{ij}$ is saved in Long-Term Tabu List (LTTL) [23], if the scout bee, that abandoned a solution, meets the $\left|f_{x_{best}} - f_{x_j}\right| < \left|f_{x_j} - f_{x_{mean}}\right|$ and $f_{x_j} > f_{x_i}$ conditions, and manages to carry the solution to a fruitful search area. The next moves in the employed and onlooker bee phase are selected randomly from ITTL and LTTL list or generated randomly with a weight probability calculated in Eq. (7), where iw_p is the intensification weight of the candidate path (ITTL, LTTL or random search) and $intscore_p$ is the intensification score that must be at least one for all paths.

$$iw_p = intscore_p \,/ \left(\sum_{p=1}^{PI} intscore_p \right) \tag{7}$$

If the selected path p improves the visited solution, $intscore_p$ is increased by one or reduced by one if the path p fails to improve the visited solution. On the other hand, if $move_{ij}$ fails to improve the visited solution, it is saved in Short-Term Tabu List (STTL) and is prohibited to be used to generate new solutions for a limited number of iterations [9].

3.3 Genetic Operator Based Artificial Bee Colony Algorithm

GA algorithm diversifies the search randomly by using cross-over and mutation operators. In this section, four types of operator are used for the path selection in the employed bee and onlooker bee phase: (i) only cross-over operator, (ii) only mutation operator, (iii) cross-over and mutation operator together, and (iv) random search operator.

Cross-over operator in GA, generates two child solutions by taking the random elements of parent solutions x_i and x_k. However, we need the cross-over operator in ABC algorithm to generate one child solution for each x_i. Chaurasia et al. [18] proposed a genetic operator-based ABC algorithm, in which the multi-point insert method replaces the cross-over operator by generating one child solution from two parent solutions as shown in Fig. (3). In genetic operator-based ABC (GABC) algorithm, multi-point insert method is used as a path to generate the new solution.

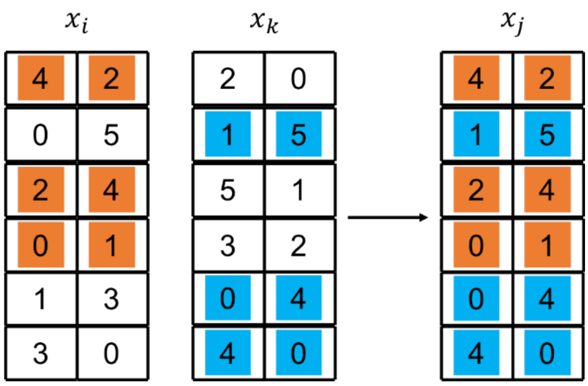

Figure 3: Multi-Point Insert Method

The path that uses the mutation operator to swap only two item-elements of the x_i to generate the new solutions. The third path is to use multi-point insert method and mutation operator respectively to generate the new solution x_j. The last path is the random search same as in MIABC algorithm. The $intscore_p$ and iw_p of the paths are calculated as in Eq. (7) to select the one of paths by roulette wheel probability.

In the GABC algorithm, the solution search paths; (i) random food source replacement rfs using random search and (ii) elite food source guided replacement efs using the interaction between the abandoned solution and the elite food sources " e ", are weighted to improve the diversification aspect in the re-scout bee phase [24].

$$dw_p = divscore_p \,/ \left(\sum_{p=1}^{PD} divscore_p \right) \tag{8}$$

If the fitness value of the generated solution, that replaces the abandoned one, meets the $\left|f_{x_{best}} - f_{x_j}\right| < \left|f_{x_j} - f_{x_{mean}}\right|$ and $f_{x_j} > f_{x_i}$ conditions, the diversification score of the path $divscore_p$ is increased by one. The search path in the re-scout bee phase is selected by roulette wheel probability calculated as in Eq. (8), where dw_p is the diversification weight of the candidate path (rfs or efs).

4 COMPUTATIONAL RESULTS AND DISCUSSION

The basic ABC, MIABC and GABC algorithms that are developed for 3D-BPP have been coded in MATLAB R2016 version software. All experimental runs are performed by the CPU that has 4 GB RAM and 3.10 GHz processors using Windows 7 operating system. Developed algorithms are tested on a randomly generated dataset according to the random instance generator used by [1]. The generated data set includes following five types of random items to allocate into the bins with uniform dimensions where $D = W = H = 100$.

Each data set class consists of five item type $k = (1, \ldots, 5)$, items of type k are chosen with probability 60%, and the rest four types are chosen with probability 10%, so the developed algorithms are tested on three classes of data sets consisting of 25 sub-problems. Each data set class considers the number of items to be placed as 20, 50, and 100, respectively.

Parameter variables of ABC algorithm, the size of population SN, total number of evaluations $Eval$, and the failure limit of each bee L determine framework of the solution search process, where

n is the number of items to be placed into bins. In addition, the number of elite bees e determines how many problem solutions are capable of attracting others in the population to reduce the duration of convergence of the GABC algorithm. The size of the STTL, LTTL and ITTL determines the number of search moves saved into the list that guides the search using useful moves and avoiding prohibited ones. The parameter values were set as seen in Table 1.

Table 1: Parametric Details of the Algorithmic Configuration

Algorithm	SN	$Eval$	e	m	L
ABC	50	$5n^2$	–	–	25
MIABC	50	$5n^2$	–	$n \times SN$	25
GABC	50	$5n^2$	10	–	25

The basic ABC algorithm involves generating the first population from one individual and the same initial populations are randomly generated for sub-problems of each data set class. The graphs in Fig. (4) shows the search history of average obtained values of 25 sub-problems for each data set class in the average bin usage ratio (BUR) obtained by the basic ABC algorithm and proposed approaches, and Table 2 shows performance of the three approaches on the generated data set. The search history of each class is statistically analyzed for the basic ABC and the proposed approaches by using one-way ANOVA and Fisher's Least Significant Difference (LSD) post hoc test.

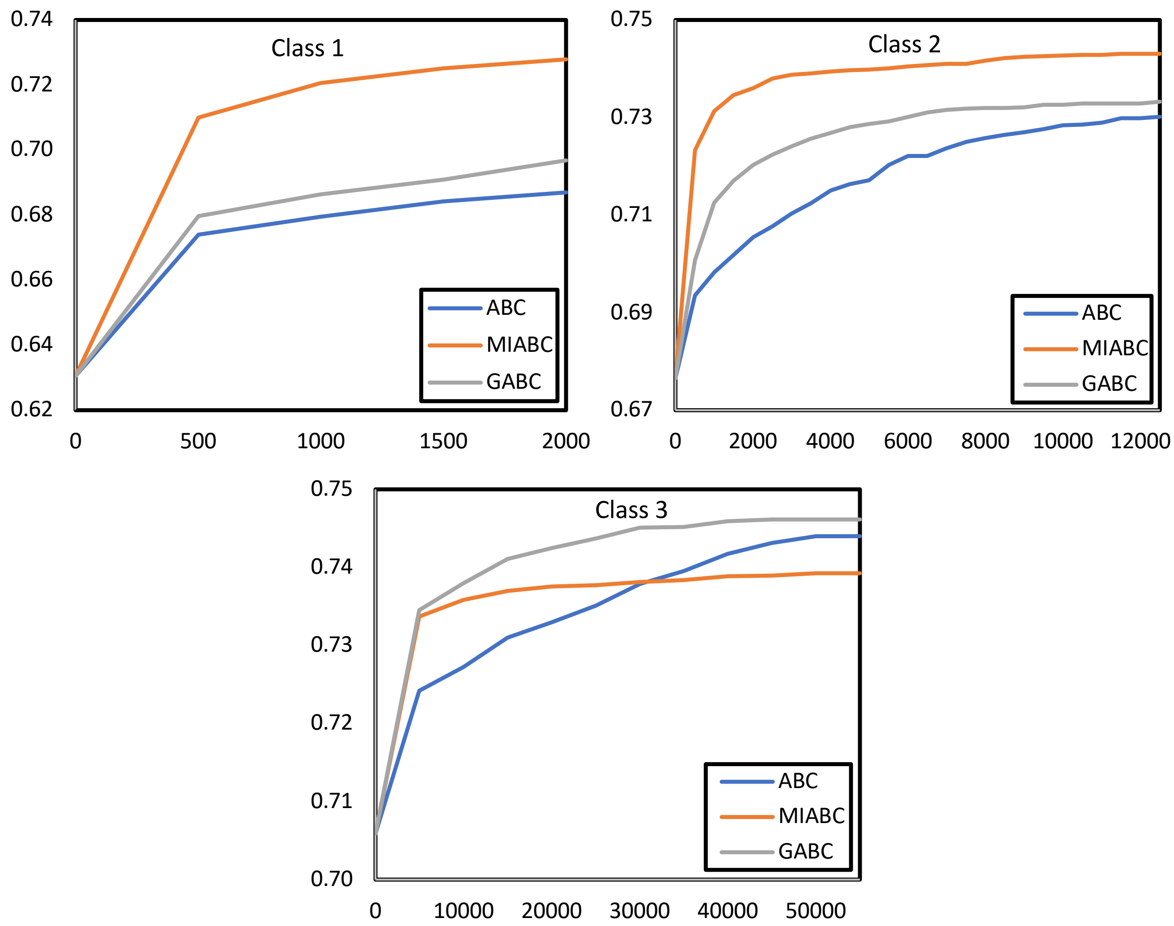

Figure 4: Search History of Three Classes of Data Sets in BUR Value

Fig. (4) depicts the change of BUR values over the evaluations. The characteristics of the three approaches vary according to the data set class they are applied to. The memory mechanism and genetic operators affect the basic ABC algorithm significantly as seen in the graphs in Fig. (4), where the three approaches are significantly different ($p < .05$) for all three data set classes. The MIABC algorithm is superior to the GABC algorithm with a difference of 3% for the Class 1 data set and 1% for the Class 2 data set, as seen in Table 2. However, as the complexity of the problem increases with the number of items to be placed, the MIABC algorithm loses its effectiveness for Class 3 data set and even lags the basic ABC algorithm. The memory mechanism shows its effectiveness in early stage of the search with significant outcomes; however, the search is limited in the long run by the memory mechanism itself. The results of the MIABC algorithm and the search process show that the memorized movements can mislead the search. As the number of prohibited moves is increased, the possibility of exploring new solutions in the local area is also restricted. On the other hand, the memorized moves in ITTL and LTTL are not as effective as expected to lead the solution search to fruitful areas in the search space.

Table 2: Average Performance of the Basic ABC Algorithm and Proposed Approaches Over Average BUR Values for Each Data Set Class

	ABC	MIABC	GABC
Class 1 (n=20)	0.6870	**0.7279**	0.6968
Class 2 (n=50)	0.7302	**0.7431**	0.7332
Class 3 (n=100)	0.7440	0.7392	**0.7461**

GABC algorithm manages to improve the basic ABC algorithm for all data set classes. However, although the GABC algorithm obtains significantly better results than the ABC algorithm in the early stage, the difference between two approaches closes over time. Unlike the MIABC algorithm, the GABC algorithm spreads the solution search through evaluations and provides a continuous improvement in the BUR value. Besides, the change in the problem complexity does not affect the GABC algorithm as much as the MIABC algorithm.

As a result, the capacity of memory mechanism is limited for three-dimensional bin packing problems and the memory integrated ABC algorithm converges prematurely when it is applied on high complex problems. However, the genetic operator integrated ABC algorithm avoids getting stuck at the local optimum with its strong diversification aspect, and the genetic operator-based reinforcement more useful than the memory mechanism in the long run.

5 CONCLUSION

The ABC algorithm is commonly used for numeric optimization problems. However, a binary optimization-based ABC algorithm has recently been applied to the three-dimensional bin packing problems, which is a kind of knapsack problem and one of the main problems of transport systems. In most ABC algorithms applied knapsack problem studies, a number of dimensions are considered as the number of parameters for each item, while in this study three-dimensional items are regarded as orthogonal objects to be placed in a three-dimensional knapsack called as a container.

The ABC algorithm is a powerful and efficient algorithm for numerical optimization with a combination of intensification and diversification aspects. However, enhanced ABC algorithms are required to solve complex problems such as 3D bin packing problems. This study focuses on enhancing the search mechanisms that are used in local search and global search to develop a robust ABC algorithm for container loading problems. A memory mechanism is used to avoid repetitive item placement solutions and to benefit from the fruitful ones in local search, while the genetic operators are integrated into the basic ABC algorithm to expand the global search area in order to discover potential better solutions. Reinforcement approaches that improve different aspects of the basic ABC algorithm are analyzed separately to understand their effect on the algorithm.

As a result, this study proposes a memory-integrated ABC algorithm to meticulously select useful search steps in local search, and a genetic operator-based ABC algorithm to intelligently generate the next search steps in global search inspired by efficient solutions. The results show that using the memory mechanism is more effective in the short run. However, it loses its effectiveness in the long run and cannot be applied to the 3D bin packing problems with high complexity, while the genetic operator-based ABC algorithm provides better solutions in the long run and is more robust than MIABC algorithm, in high complexity.

The BUR values obtained from the proposed algorithms can be improved in future studies, focusing on bin packing heuristics, based on decoding the complete random packing sequence in this study. The packing sequences can be generated that place similar items in blocks, layers or stacks to reduce spaces in bins. On the other hand, a joint hybrid algorithm, that uses both memory mechanism and genetic operators, can be developed to observe the effects of the proposed approaches when they are used together as reinforcement approaches for the basic ABC algorithm.

REFERENCES

[1] S. Martello, D. Pisinger, and D. Vigo, "The three-dimensional bin packing problem," *Operations research*, vol. 48, no. 2, pp. 256–267, 2000.

[2] N. Nepomuceno, P. Pinheiro, and A. L. v Coelho, "Tackling the Container Loading problem: A hybrid approach based on Integer Linear Programming and Genetic Algorithms," in *Evolutionary Computation in Combinatorial Optimization, Proceedings*, vol. 4446, C. Cotta and J. VanHemert, Eds. 2007, pp. 154-+.

[3] T. Tian, W. B. Zhu, A. Lim, and L. J. Wei, "The multiple container loading problem with preference," *European Journal of Operational Research*, vol. 248, no. 1, pp. 84–94, Jan. 2016, doi: 10.1016/j.ejor.2015.07.002.

[4] K. Kang, I. Moon, and H. F. Wang, "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, 2012, doi: 10.1016/j.amc.2012.07.036.

[5] L. Junqueira and R. Morabito, "Heuristic algorithms for a three-dimensional loading capacitated vehicle routing problem in a carrier," *Computers & Industrial Engineering*, vol. 88, pp. 110–130, 2015, doi: 10.1016/j.cie.2015.06.005.

[6] A. Moura and A. Bortfeldt, "A two-stage packing problem procedure," *International Transactions in Operational Research*, vol. 24, no. 1–2, pp. 43–58, 2017, doi: 10.1111/itor.12251.

[7] S. Sundar and A. Singh, "A swarm intelligence approach to the quadratic multiple knapsack problem," 2010, pp. 626–633.

[8] C. Ozturk, E. Hancer, and D. Karaboga, "A novel binary artificial bee colony algorithm based on genetic operators," *Information Sciences*, vol. 297, pp. 154–170, 2015.

[9] T. Bayraktar, M. E. Aydin, and M. Dugenci, "A memory-integrated artificial bee algorithm for 1-D bin packing problems," in *9th International Symposium on Intelligent Manufacturing and Service Systems*, 2014, pp. 1023–1034.

[10] H. Gehring and A. Bortfeldt, “A genetic algorithm for solving the container loading problem,” *International transactions in operational research*, vol. 4, no. 5–6, pp. 401–418, Nov. 1997, doi: 10.1111/j.1475-3995.1997.tb00095.x.

[11] A. Bortfeldt and H. Gehring, “A hybrid genetic algorithm for the container loading problem,” *European Journal of Operational Research*, vol. 131, no. 1, pp. 143–161, 2001, doi: 10.1016/s0377-2217(00)00055-2.

[12] Y. Wu, W. K. Li, M. Goh, and R. de Souza, “Three-dimensional bin packing problem with variable bin height,” *European Journal of Operational Research*, vol. 202, no. 2, pp. 347–355, 2010, doi: 10.1016/j.ejor.2009.05.040.

[13] A. Bortfeldt and H. Gehring, “Applying tabu search to container loading problems,” in *Operations Research Proceedings 1997*, Springer, 1998, pp. 533–538.

[14] J. M. Liu, Y. Yue, Z. R. Dong, C. Maple, and M. Keech, “A novel hybrid tabu search approach to container loading,” *Computers & Operations Research*, vol. 38, no. 4, pp. 797–807, 2011, doi: 10.1016/j.cor.2010.09.002.

[15] D. Mack, A. Bortfeldt, and H. Gehring, “A parallel hybrid local search algorithm for the container loading problem,” *International Transactions in Operational Research*, vol. 11, no. 5, pp. 511–533, Sep. 2004, doi: 10.1111/j.1475-3995.2004.00474.x.

[16] W. B. Zhu, H. Qin, A. Lim, and L. Wang, “A two-stage tabu search algorithm with enhanced packing heuristics for the 3L-CVRP and M3L-CVRP,” *Computers & Operations Research*, vol. 39, no. 9, pp. 2178–2195, 2012, doi: 10.1016/j.cor.2011.11.001.

[17] V. Panahi and N. J. Navimipour, “Join query optimization in the distributed database system using an artificial bee colony algorithm and genetic operators,” *Concurrency and Computation: Practice and Experience*, p. e5218, 2019.

[18] S. N. Chaurasia, S. Sundar, and A. Singh, “Hybrid metaheuristic approaches for the single machine total stepwise tardiness problem with release dates,” *Operational Research*, vol. 17, no. 1, pp. 275–295, 2017.

[19] F. A. N. Chengli, F. U. Qiang, L. Guangzheng, and X. Qinghua, “Hybrid artificial bee colony algorithm with variable neighborhood search and memory mechanism,” *Journal of Systems Engineering and Electronics*, vol. 29, no. 2, pp. 405–414, 2018.

[20] D. Karaboga, “An idea based on honey bee swarm for numerical optimization,” Technical report-tr06, Erciyes university, engineering faculty, computer engineering department, 2005.

[21] S. Sundar, A. Singh, and A. Rossi, “An artificial bee colony algorithm for the 0–1 multidimensional knapsack problem,” 2010, pp. 141–151.

[22] M. S. Kiran and M. Gündüz, “XOR-based artificial bee colony algorithm for binary optimization,” *Turkish Journal of Electrical Engineering & Computer Sciences*, vol. 21, no. Sup. 2, pp. 2307–2328, 2013.

[23] F. Glover, “Tabu search—part I,” *ORSA Journal on computing*, vol. 1, no. 3, pp. 190–206, 1989.

[24] P. K. Singhal, R. Naresh, and V. Sharma, “A modified binary artificial bee colony algorithm for ramp rate constrained unit commitment problem,” *International Transactions on Electrical Energy Systems*, vol. 25, no. 12, pp. 3472–3491, 2015.