IJCCS (Indonesian Journal of Computing and Cybernetics Systems)
Vol.16, No.1, January 2022, pp. 55~66
ISSN (print): 1978-1520, ISSN (online): 2460-7258
DOI: https://doi.org/10.22146/ijccs.70677

Fast Non-dominated Sorting in Multi Objective Genetic Algorithm for Bin Packing Problem

Muhammad Bintang Bahy*[1], Aina Musdholifah[2]
[1]Undergraduate Program of Computer Science, FMIPA UGM, Yogyakarta, Indonesia
[2]Department of Computer Science and Electronics, FMIPA UGM, Yogyakarta, Indonesia
e-mail: ***[1]muhammad.bintang.bahy@mail.ugm.ac.id**, [2]aina_m@ugm.ac.id

Abstrak

Permasalahan bin packing adalah sebuah permasalahan di mana barang dengan volume dan dimensi berbeda dimasukkan ke dalam sebuah wadah sehingga volume barang yang dimasukkan maksimal. Permasalahan bin packing multi objektif adalah permasalahan yang lebih umum ditemukan di kehidupan sehari-hari, karena yang diperhatikan dalam pengepakan biasanya tidak hanya volume.

Pada penelitian ini diajukan sebuah algoritma genetika multi objektif untuk menyelesaikan permasalahan bin packing multi objektif. Algoritma genetika yang diajukan menggunakan metode non-dominated sorting dan crowding distance untuk mendapatkan solusi yang terbaik untuk tiap objektifnya dan menghindari adanya bias. Algoritma kemudian diuji dengan beberapa kelas uji yang menyatakan kombinasi ukuran barang dan wadah yang berbeda.

Dari hasil pengujian yang dilakukan didapatkan bahwa algoritma yang diajukan dapat menemukan beberapa solusi yang merupakan kandidat solusi terbaik untuk tiap objektif. Didapatkan juga bagaimana korelasi tiap objektif pada populasi.

Kata kunci*—Algoritma genetika, bin packing problem, multi objektif, multi solusi, non-dominated sorting*

Abstract

The bin packing problem is a problem where goods with different volumes and dimensions are put into a container so that the volume of goods inserted is maximized. The problem of multi-objective bin packing is a problem that is more commonly found in everyday life, because what is considered in packing is usually not only volume.

In this research, a multi-objective genetic algorithm is proposed to solve the multi-objective bin packing problem. The proposed genetic algorithm uses non-dominated sorting and crowding distance methods to get the best solution for each objective and to avoid bias. The algorithm is then tested with several test classes that represent different combinations of item and container sizes.

From the results of the tests carried out, it was found that the proposed algorithm can find several solutions which are the best candidate solutions for each objective. Also found how the correlation of each objective in the population.

Keywords—*Genetic algorithm, bin packing problem, multi objective, multi solution, non-dominated sorting*

Received November 26th,2021; Revised January 31th, 2022; Accepted January 31th, 2022

1. INTRODUCTION

The bin packing problem is a problem where goods with different volumes and dimensions are provided, then these goods must be put into a number of containers (bins) so as to maximize the volume used [1]. There are many variations of the bin packing problem, for example 2D packing, linear packing, and 3D packing. This bin packing problem in computational complexity theory belongs to the NP-Hard problem and is difficult to solve [2].

There are many approaches used by various researchers in solving the bin packing problem with a single objective. [3] solved the multiple bin packing problem with a single objective using the mixed genetic algorithm method with the heuristic packing method using the deepest bottom left first. [4] proposed an algorithm to solve the three-dimensional multiple bin packing problem. The proposed method is to use mixed integer programming and tabu search combined with sub-volume based heuristics.

There are research that uses evolutionary algorithm to solve the multi objective bin packing problem. The existence of this multi-objective problem is due to the fact that in everyday life, in arranging goods into containers, sometimes what is taken into account is not only the use of volume, it can also be taken into account the weight or value of the goods. In a study conducted by [5], a multi-objective genetic algorithm was used to determine the solution for the arrangement of goods in containers. In the study by [5], three objective functions were used, namely the number of goods, the weight of the goods, and the volume of goods. [6] solved multi objective bin packing problem using discrete particle swarm optimization method to find the order and rotation of packing goods, while binary space partitioning was used for container arrangement. [7] conducted a study to solve the single container loading problem with two objective functions, namely the value of goods and the use of space in the container. Here [7] using a multi-objective multi-population genetic algorithm and supported by a fuzzy logic controller to optimize the genetic algorithm parameters used. [8] proposed a solution using particle swarm optimization. The objective that is trying to be solved is to minimize the use of containers, and optimize the center of gravity according to preferences.

Some research did not use the evolutionary algorithm to solve multi objective bin packing problem instead using non heuristic approach. [9] carried out research using the greedy method, using a layer-based approach. The objective function that is taken into account is the number of bins used to minimize and balance the weight of each bin. [10] uses the greedy bottom left fill and bottom-up left-justified methods. The objective function that is calculated is the same as the research by [9], namely the number of bins used and balancing the weight of each bin.

In [11] a genetic algorithm with non-dominated sorting method was found that can be used to obtain a set of solutions that are superior to other solutions of a multi-objective problem. This set of solutions is known as non-dominated solutions or Pareto optimal solutions. By using this non-dominated sorting method, there will be no bias in the selection process and the results of the genetic algorithm are in the form of a set of solutions (Pareto optimal solutions) not just one solution.

In this study, a genetic algorithm with a non-dominated sorting method was used which was adopted from the research of [12] which is the development of research by [11]. The method used is coupled with a strategy called crowding distance which makes the solutions on the Pareto front more evenly distributed. The objective function used in this research is to maximize the weight of the goods, maximize the use of space, and balance the weight of the bin.

2. METHODS

In this section, we will discuss how the proposed algorithm works. It will also explain how the algorithm can produce a solution for each objective.

2.1 Algorithm Design

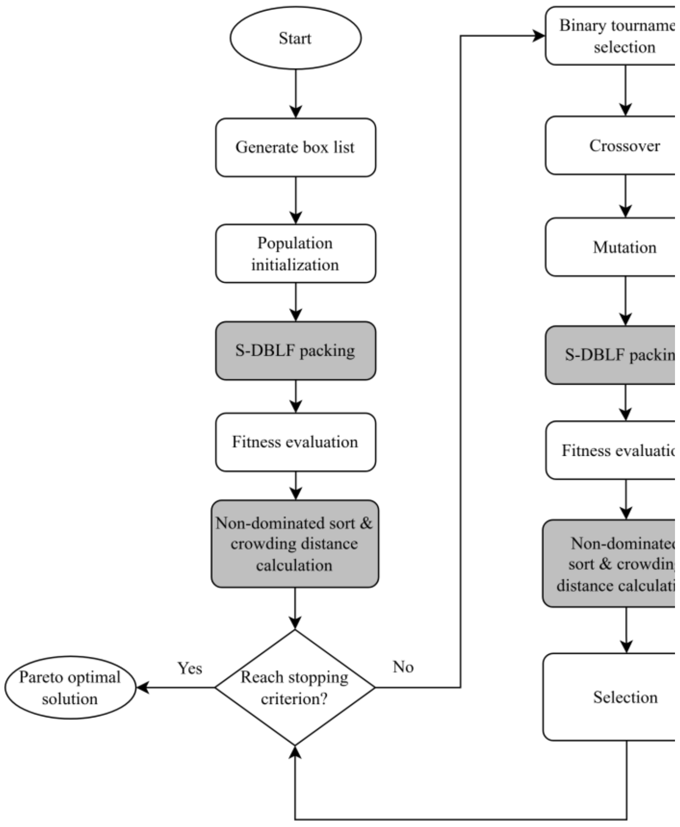
Start
Generate box list
Population initialization
S-DBLF packing
Fitness evaluation
Non-dominated sort & crowding distance calculation
Reach stopping criterion?
Yes
No
Pareto optimal solution
Binary tourname selection
Crossover
Mutation
S-DBLF packin
Fitness evaluatic
Non-dominatec sort & crowdin distance calculati
Selection

Figure 1 Algorithm flow chart

Figure 1 above is a flow chart of the algorithm that will be used in this study. In this study, the input of the algorithm is a box list that is generated randomly, then the output of the algorithm is a set (pareto optimal solution) which contains the order of packing of the box list into a bin. The algorithm built in this study uses a genetic algorithm. The next section will explain in more detail step by step the algorithm that will be used.

2. 1.1 Box list generation

The box list generation is divided into several classes , each of which has a different percentage of the number of small, medium, and large boxes described in Table 1. l_i,w_i,h_i,$\psi_i$denotes the length, width, height, and weight of the box. The box list will be generated randomly with the percentage rules for the number of types of box type in Table 2.

Table 1 Box type

Size	Small	Medium	Large
l_i,w_i,h_i	(0, 5]	(5, 10]	(10, 15]
ψ_i	(0, 5]	(10, 20]	(30, 45]

Table 2 Box class rules

Class	1	2	3	4	5	6	7
Small Box Type	100%	0%	0%	33%	50%	50%	0%
Medium Box Type	0%	100%	0%	33%	50%	0%	50%
Large Box Type	0%	0%	100%	33%	0%	50%	50%

2. 1.2 Chromosome coding

If n is the number of boxes, then the solution is encoded with a representation in the form of a string s=s_1,s_2,...,s_n,s_{n+1},s_{n+2},...,s_{2n} . Genes in the sequence s_1 to $s_n$contain number markers in the box, while the sequence states the order of packing into bins. While the genes in the sequence s_{n+1} to s_{2n}represent the rotation of the box in the same order. The rotation value is 1 if the box is rotated 90° with respect to the horizontal plane, and is 0 if no rotation is carried out. Table 3 is an example of chromosome coding.

Table 3 Chromosome coding example

Packing order	2	3	1	4	6	5	9	8	7	10
Rotation	1	1	0	1	0	0	1	1	0	1

2. 1.3 Population Initialization

The initialization process is carried out by generating a solution with a random packing sequence and rotation.

2. 1.4 *S-DBLF Packing*

Simple Deepest Bottom Left with Fill (S-DBLF) is a packing method that has been proposed by [13]. In the S-DBLF method, it tries to pack into the first available and valid place with the DBL sequence. Packing can be said to be valid if it meets the following conditions.

a) Box does not exceed the size limit of the bin.
b) Boxes do not intersect with other boxes.
c) The box does not hang, meaning all the space under the box to be placed must be filled by another box.

Figure 2 is the pseudocode of the S-DBLF packing method.

```
procedure S-DBLF packing:
begin
initialize position set P;
        for each i ∈ B do
                check := false;
                sort position set P according to DBL order;
                for each j ∈ P do
                        if i can be packed into j do
                                check := false;
                                set j as packing position of i;
                                update position set P;
                                break;
                        end if
                end for
                if check == false then break;
        end for
end
```

Figure 2 S-DBLF pseudocode

2. 1.5 Objective Function and Fitness Evaluation

In this study, three objective functions and fitness function will be used. The following is an explanation of the three objective functions and the fitness function. Below is an explanation of the variables that will be used in the calculation of the objective function and the fitness function.

L,W	Length and width of bin.
H	Packing height.
l_i,w_i,h_i,ψ_i	Length, width, and height of box i
$posX_i$,$posY_i$,$posZ_i$	Deepest bottom left position in *x*, *y*, and *z*

	axis of box i
x_i	A binary number that is 1 if the box i is in the bin, and 0 if it is not.
n	Box count.
c_x	The bin's center of gravity is on the x-axis.
c_y	The bin's center of gravity is on the y-axis.

1. Volume utilization
The fitness function and the objective function for volume optimization aim to maximize the use of space so that there is less free space in the bin. Equation 1 is the fitness function for the volume utilization. While equation 2 is an objective function for the volume utilization.

$$f_1 = \sum_{\mathrm{i}=1}^{n} x_i\, l_i w_i h_i \tag{1}$$

$$z_1 = \text{maximize}\left(\sum_{\mathrm{i}=1}^{n} x_i\, l_i w_i h_i\right) \tag{2}$$

2. Total box weight
The fitness function and objective function for the total weight of the box aims to maximize the weight of the items packed into the bin. In this fitness function, the total weight of all items that packed to the bin is calculated. Equation 3 is the fitness function for the total weight. While equation 4 is the objective function for the total weight.

$$f_2 = \sum_{\mathrm{i}=1}^{n} x_i\, \psi_i \tag{3}$$

$$z_2 = \text{maximize}\left(\sum_{\mathrm{i}=1}^{n} x_i\, \psi_i\right) \tag{4}$$

3. Bin balance
The fitness function and objective function for bin balance aim to make the center of gravity of all items that packed to the bin as close as possible to the midpoint of the bin to the horizontal plane. To calculate the proximity, a calculation using the Euclidean distance from the center of gravity of all items combined with the midpoint of the bin is used. Equations 5, 6, 7, and 8 are the fitness functions for the bin balance. While equation 9 is the objective function for bin balance.

$$c_x = \frac{\sum_{\mathrm{i}=1}^{n} x_i\, \psi_i \left(\mathrm{posX}_i + \frac{l_i}{2}\right)}{\sum_{\mathrm{i}=1}^{n} x_i\, \psi_i} \tag{5}$$

$$c_y = \frac{\sum_{\mathrm{i}=1}^{n} x_i\, \psi_i \left(\mathrm{posY}_i + \frac{w_i}{2}\right)}{\sum_{\mathrm{i}=1}^{n} x_i\, \psi_i} \tag{6}$$

$$c_z = \frac{\sum_{\mathrm{i}=1}^{n} x_i\, \psi_i \left(\mathrm{posZ}_i + \frac{h_i}{2}\right)}{\sum_{\mathrm{i}=1}^{n} x_i\, \psi_i} \tag{7}$$

$$f_3 = \sqrt{\left(\left(\frac{L}{2}\right) - c_x\right)^2 + \left(\left(\frac{W}{2}\right) - c_y\right)^2 + \left(\left(\frac{H}{2}\right) - c_z\right)^2} \tag{8}$$

$$z_3 = \text{minimize}\left(\sqrt{\left(\left(\frac{L}{2}\right) - c_x\right)^2 + \left(\left(\frac{W}{2}\right) - c_y\right)^2 + \left(\left(\frac{H}{2}\right) - c_z\right)^2}\right) \tag{9}$$

2. 1.6 Fast Non-dominated Sort (FNDS)

The fast non-dominated sorting algorithm proposed by [12], Figure 3 is the pseudocode of the fast non-dominated sorting algorithm.

```
procedure fast-non-dominated-sort(P):
P = population
ni = domination counter of solution i
Fi = i-th front
P-rank = pareto ranking
begin
        for each p ∈ P do
                Sp = Ø
                np = 0
                for each q ∈ P do
                        if p dominates q do
                                Sp = Sp ∪ {q}
                        else if q dominates p do
                                np = np + 1
                if np = 0 do
                        P-rank = 1
                        F1 = F1 ∪ {p}
        i = 1
        while Fi != Ø do
                Q = Ø
                for each p ∈ Fi do
                        for each q ∈ Sp do
                                nq = nq - 1
                                if nq = 0 do
                                        q-rank = i + 1
                                        Q = Q ∪ {q}
                i = i + 1
                Fi = Q
end
```

Figure 3 Fast non-dominated sorting pseudocode

2. 1.7 Crowding Distance Calculation

The crowding distance calculation algorithm proposed by [12]. Figure 4 is the pseudocode of the crowding distance calculation.

```
procedure crowding-distance-assignment(F):
F = a pareto front
begin
        l = |F|
        for i = 1 to l do
                F[i].distance = 0
                for each objektif m do
                        F = sort(F, m)
                F[1].distance = infinity
                F[l].distance = infinity
                for i = 2 to (l - 1) do
                        F[i].distance = F[i].distance + (F[i+1].m -
F[i-1].m) / (fmax(m) - fmin(m))
end
```

Figure 4 Crowding distance calculation pseudocode

2. 1.8 Parent Selection (Tournament Selection)

The type of tournament selection that will be used is binary tournament selection, where two chromosomes will be selected at random and then compared which one is better. Two chromosomes compared to which one has the smaller pareto front. If they are the same, the one with the larger crowding distance will be chosen.

2. 1.9 Crossover

The type of crossover used is the PMX crossover. The number of offspring resulting from this crossover process is n. Each parent pair will produce one offspring, so the crossover process will be performed n times.

2. 1.10 Mutation

The mutation process is done by generating a random value between 0-1, and then.

a) Swapping the packing order at random if random number less than 0.5.
b) Reverse the rotation of a box randomly otherwise.

2. 1.11 Generation Update

After the genetic process, $2n$ individuals will be obtained. Then n best individuals will be selected in the order as in the parent selection process.

2. 1.12 Stopping Criteria

The algorithm will be terminated when the specified maximum number of generations is reached.

2. 1.13 Solution Selection

There are 4 solutions selected from the final population, namely the solution with the largest fitness volume, the solution with the largest weight fitness, the solution with the smallest box balance fitness, and the solution with the largest total fitness normalization. Equation 10 is the for calculating normalized fitness.

$$f_{\text{norm}} = \frac{f_1}{\sum_{\text{i}=1}^{n} l_i\, w_i h_i} + \frac{f_2}{\sum_{\text{i}=1}^{n} \psi_i} + \frac{f_3 - \frac{\sqrt{L^2+\text{W}^2+\text{H}^2}}{2}}{\frac{\sqrt{L^2+\text{W}^2+\text{H}^2}}{2}} \tag{10}$$

3. RESULTS AND DISCUSSION

In this section, we will discuss the tests that have been carried out on the classes that have been described previously. In this test, algorithm performance, algorithm parameters, and the correlation between objectives will be discussed. For each class will be tested against various pairs of parameters as well as the number of boxes and bin size large or small. The small bin size is the minimum size to place 50 boxes, while the large bin size is the minimum size to place 100 boxes. Each parameter pair will be tested 10 times then averaged.

3.1 Algorithm Performance

Based on the data in Table 4 the algorithm in this study runs best according to its fitness in class 1 which contains 100% small size boxes. While the worst performance of the algorithm occurs in class 3 which contains 100% of large boxes. Classes that have large box compositions all occupy the last few rankings. Class 7 is ranked 4th with a composition of 50% medium-sized boxes and 50% large-sized boxes. Then class 4 ranks 5th with a composition of 33% small boxes, 33% medium boxes, and 33% large boxes. Class 6 is ranked 6th with a composition of 50% big boxes and 50% small boxes.

Then the other 2 classes that do not have a large box composition are ranked 2 and 3. At rank 3 there is class 5 with a composition of 50% small boxes and 50% medium boxes. Class 2 is ranked 2nd with a composition of 100% medium-sized boxes. Class ranking if it is not viewed from its fitness, but from each objective, the ranking obtained is also similar to a difference of 1-2 ranks compared to the ranking for its fitness.

Table 4 Average objective score for each class

Class	Average Fitness	Average Volume Percentage	Average Center of Mass	Average Density
1	2.71172402221213	0.87716516385453	0.151958186541507	0.143035417790328
2	2.48244756200641	0.818417399588127	0.492967490751399	0.0327068522914782
3	2.29893548164599	0.759072594519474	1.52388922083252	0.0180606920020255
4	2.36125433833604	0.769041349289125	0.935078296442724	0.0226068402123578
5	2.42686603714604	0.812025304831673	0.83973258910332	0.037875160208339
6	2.33319489465164	0.789263348939289	1.63140833396836	0.0192526904079276
7	2.41978143075527	0.829655733297394	0.656049039699358	0.0207811766493336

If the performance of the algorithm is viewed from the number of boxes and bin size pairs in Table 5 for fitness, volume, and density in the 50-small and 100-big pairs the results are not much different. The difference is in the center of mass, for the 100-big pair it is worse than the 50-small pair. As for the 100-small pair, it is better for all objectives, this shows that the algorithm has succeeded in selecting the box so that the objective value is better. As for the fitness in the 100-small pair, it tends to be smaller because the fitness divisor calculation on the weight objective is dynamic.

Table 5 Average objective value for each pair of box count and bin size

Box Count	Bin Size	Average Fitness	Average Volume Percentage	Average Center of Mass	Average Density
50	small	2.66861924034999	0.775656668459268	0.73843626014872	0.0212326581762687
100	big	2.61081513124255	0.778747216076289	1.3097258297426	0.0217711861219594
100	small	1.99909139500773	0.872356842546794	0.596841983013724	0.0238808231659094

3.2 Genetic Algorithm Parameters

When viewed from the type of class in Table 6, for all classes, both mode and median, the best population size data is 100 individuals. For the number of generations in both the mode and median parameter data, the best is 500 generations with different classes being class 5 with 300 generations. Of the 7 classes, there are 6 classes that have a mode of the number of generations of 500 and also a median of 500 or 450 which indicates half or more of the data has a number of generations of 500 because the number of generations of 500 is the largest value of the 5 values of the number of generations tested. As for the mutation probability parameter, the best parameter for the majority of classes is 10%. Of the 7 classes, there are 5 classes with a mode of 10% and also a median of 10% which indicates half or more of the data has a mutation probability of 10%.

Table 6 The best genetic algorithm parameters by class

Class	Modus			Median		
	Population Size	Generation	Mutation Probabilty	Population Size	Generation	Mutation Probabilty
1	100	500	5	100	500	7.5
2	100	500	10	100	500	10
3	100	500	10	100	500	10
4	100	500	10	100	500	10
5	100	300	5	100	400	5
6	100	500	10	100	500	10
7	100	500	10	100	450	10

When viewed from the type of solution in Table 7, both the mode and the median have the same value. To get the type of solution with the best fitness, volume, and center of mass, genetic algorithm parameters are used for population size of 100, number of generations of 500, and mutation probability of 10%. To get the type of solution with the best weight, the parameters of the genetic genetic algorithm are population size 100, the number of generations is 500, and the mutation probability is 5%. For each type of solution, the number of generations parameter has a mode of 500 and a median of 500 which indicates half or more of the data has a number of generations of 500. Meanwhile, of the 4 types of solutions, there are 3 types of solutions that have a mutation probability mode of 10% and a median of 10% also which indicates half or more than the data has a mutation probability of 10%.

Table 7 The best genetic algorithm parameters based on the solution type

Solution Type	Modus			Median		
	Population Size	Generation	Mutation Probabilty	Population Size	Generation	Mutation Probabilty
fitness	100	500	10	100	500	10
volume	100	500	10	100	500	10
weight	100	500	5	100	500	5
center of mass	100	500	10	100	500	10

Table 8 shows the best genetic algorithm parameters when viewed based on pairs of box count and bin size. Almost all of the data obtained for the mode and median use population size parameters of 100, number of generations of 500 and mutation probability of 10%. The median value equal to the mode indicates that half or more of the data use the genetic algorithm's parameters. There is only one median difference in the 100-big pair with 400 generations.

Table 8 Genetic algorithm parameters for each pair of box count and bin size

Box Count	Bin Size	Modus			Median		
		Population Size	Generation	Mutation Probabilty	Population Size	Generation	Mutation Probabilty
50	small	100	500	10	100	500	10
100	big	100	500	10	100	400	10
100	small	100	500	10	100	500	10

3.3 Correlation Between Objectives

Table 9 shows the correlation between the objective pairs by class. In class 4-7 the correlation for the volume-weight pair is close to 0 which indicates there is no correlation between the two objectives. Whereas in the objective pair of volume-center of mass and weight-center of mass there is a positive correlation which means that the greater the volume/weight value, the greater the center of mass value. This positive correlation means that if the volume/weight is getting bigger (improved) then the objective value of the center of mass is getting bigger which means it is getting worse because the objective of the center of mass is to minimize its value. When viewed in class 1-3 there is a positive correlation in the objective pair of volume-weight, whereas in the objective pair of volume-center of mass and weight-center of mass the same as in class 4-7 there is a positive correlation. It can also be noted that in class 1-3 the composition of the box consists of only one type of box, while in class 4-7 the composition is mixed.

Table 9 Average correlation between objective pairs by class

Class	Average Correlation		
	Volume-Weight	Volume-Center of Mass	Weight-Center of Mass
1	0.320652782220033	0.733352027377626	0.604236122614511
2	0.247556934983511	0.683192431069511	0.7184163420218
3	0.526396116180746	0.804763271855138	0.798589719150745
4	0.0545362846574056	0.638709584620526	0.485946235119459
5	0.0233145683318221	0.485921758011147	0.597245619584405
6	-0.0189107745044873	0.544167487826542	0.518894180091024
7	0.00476752072351774	0.641608540664168	0.56899275000019

If the correlation between the objectives is seen from the data in Table 10 which shows the correlation between the objectives based on the type of solution, the correlation values generated for each pair of objectives tend not to differ much. For the three objective pairs there is a positive correlation, where the volume-weight pair has a correlation value that is not as large as the other pairs.

Table 10 Average correlation between objective pairs by solution type

Best Individual Type	Average Correlation		
	Volume-Weight	Volume-Center of Mass	Weight-Center of Mass
fitness	0.165363593026624	0.632974723938897	0.622243934219339
volume	0.156549338232462	0.654358870264465	0.597110214844801
weight	0.140873429841239	0.636367076817193	0.585069795072851
center of mass	0.198360557124406	0.664994063438903	0.647757541730327

Table 11 shows the correlation between the objective pairs when viewed from the number of boxes and bin size pairs. For the volume-center-of-mass pairs for all pairs of box count and bin size there is a positive correlation, the objective pair of weight-center of mass also has positive correlation. Meanwhile, for the volume-weight objective pair, there is only a positive correlation in the 50-small and 100-big pairs, while in the 100-big pair the correlation value is close to zero. This might happen because the box selection has been optimized so that the volume and weight objectives are no longer correlated, and what remains is the box settings in the bin.

Table 11 Average correlation between objective pairs based on pairs of box count and bin size

Box Count	Bin Size	Average Correlation		
		Volume-Weight	Volume-Center of Mass	Weight-Center of Mass
50	small	0.245345622347975	0.611516971621104	0.678986501388532
100	big	0.255898559276253	0.712128134288087	0.608605393677066
100	small	-0.0144805353201512	0.615051485827796	0.549876783980437

4. CONCLUSIONS

From the research that has been done, several conclusions were obtained. The algorithm proposed in this study succeeded in finding several optimal solutions by considering three different objectives. The optimal solution for each objective is found and there is no bias towards a single objective in the final population. The algorithm works best on small box types and worst on large boxes for cube-shaped bins. The best genetic algorithm parameters in general to produce the most optimal solution are population size of 100, number of generations of 500, and mutation probability of 10%. There is a positive correlation in the objective pair of volume-center of mass and weight-center of mass. Meanwhile, the volume-weight pair is also positively correlated but its value tends to be closer to zero.

In this research there is still much that can be done to improve it in the future. First, the algorithm can be tested on bins with shapes other than cubes. Then it can be tested for mutation probability above 10%. After that, the weight objective can be given a maximum value limit that can be entered into the bin so that when calculating fitness it can be divided by a non-dynamic divisor so that the test is more measurable.

REFERENCES

[1] R. E. Korf, “A new algorithm for optimal bin packing,” *Proc. Natl. Conf. Artif. Intell.*, pp. 731–736, 2002.

[2] S. Martello, D. Pisinger, and D. Vigo, “Universita degli Studi di Bologna The Three-Dimensional Bin Packing Problem The Three-Dimensional Bin Packing Problem,” *Rev. Politécnica*, vol. 29, 2010.

[3] X. Feng, I. Moon, and J. Shin, “Hybrid genetic algorithms for the three-dimensional multiple container packing problem,” *Flex. Serv. Manuf. J.*, vol. 27, no. 2–3, pp. 451–477, 2015, doi: 10.1007/s10696-013-9181-8.

[4] Z. Jin, T. Ito, and K. Ohno, “The three-dimensional bin packing problem and its practical algorithm,” *JSME International Journal, Series C: Mechanical Systems, Machine Elements and Manufacturing*, vol. 46, no. 1. pp. 60–66, 2003, doi: 10.1299/jsmec.46.60.

[5] P. A. Wibowo, “Algoritme Genetika Multi Objektif untuk Mengoptimalkan Penyusunan Barang dalam Kontainer,” *Tesis, Univ. Gadjah Mada*, 2017.

[6] C. He, Y. B. Zhang, J. W. Wu, and C. Chang, “Research of three-dimensional container-packing problems based on discrete particle swarm optimization algorithm,” *Proc. Int. Symp. Test Meas.*, vol. 2, pp. 425–428, 2009, doi: 10.1109/ICTM.2009.5413015.

[7] J. N. Zheng, C. F. Chien, and M. Gen, “Multi-objective multi-population biased random-key genetic algorithm for the 3-D container loading problem,” *Comput. Ind. Eng.*, vol. 89, pp. 80–87, Nov. 2015, doi: 10.1016/j.cie.2014.07.012.

[8] D. S. Liu, K. C. Tan, S. Y. Huang, C. K. Goh, and W. K. Ho, “On solving multiobjective bin packing problems using evolutionary particle swarm optimization,” *Eur. J. Oper. Res.*, vol. 190, no. 2, pp. 357–382, Oct. 2008, doi: 10.1016/j.ejor.2007.06.032.

[9] J. Hasan, “Multi-objective 3D bin-packing problem,” *2019 8th Int. Conf. Model. Simul. Appl. Optim.*, pp. 1–5, 2019.

[10] J. Kaabi, Y. Harrath, H. E. Bououdina, and A. T. Qasim, “Toward smart logistics: A new algorithm for a multi-objective 3D bin packing problem,” *IET Conf. Publ.*, vol. 2018, no. CP747, pp. 4–8, 2018, doi: 10.1049/cp.2018.1384.

[11] N. Srinivas and K. Deb, “Muiltiobjective Optimization Using Nondominated Sorting in Genetic Algorithms,” *Evol. Comput.*, vol. 2, no. 3, pp. 221–248, 1994, doi: 10.1162/evco.1994.2.3.221.

[12] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, “A fast and elitist multiobjective genetic algorithm: NSGA-II,” *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, 2002, doi: 10.1109/4235.996017.

[13] K. Kang, I. Moon, and H. Wang, “A hybrid genetic algorithm with a new packing strategy for the three-dimensional bin packing problem,” *Appl. Math. Comput.*, vol. 219, no. 3, pp. 1287–1299, 2012, doi: 10.1016/j.amc.2012.07.036.