Hybrid Heuristic Algorithms for the Multiobjective Load Balancing of 2D Bin Packing Problems

Muhammed Beyaz, Tansel Dokeroglu and Ahmet Cosar

Abstract 2D Bin packing problem (2DBPP) is an NP-hard combinatorial optimization problem. Multiobjective versions of this well-known industrial engineering problem can occur frequently in real world application. Recently, Hybrid Evolutionary Algorithms have appear as a new area of research with their ability to combine alternative heuristics and local search mechanisms together for higher quality solutions. In this study, we propose a set of novel multiobjective hybrid genetic and memetic algorithms that make use of the state-of-the-art metaheuristics and local search techniques for minimizing the number of bins while also maintaining the load balance. We analyze the optimization time and the resulting solution quality of the proposed algorithms on an offline 2DBPP benchmark problem set with 500 instances. Using these results of exhaustive experiments, we conclude that the proposed hybrid algorithms are robust with their ability to obtain a high percentage of the optimal solutions.

1 Introduction

The two-dimensional Bin Packing Problem (2DBPP) consists of planning a set of rectangular items into a fixed width and height 2D bins orthogonally, without overlapping, while minimizing the number of bins [1–5]. The 2DBPP is an intractable optimization problem and widely faced during the industrial manufacturing processes. Proposed algorithms find a pareto-optimal solution for both the minimal number of bins with the most efficiently load-balanced placing of the rectangular items. rotating

M. Beyaz · T. Dokeroglu (✉) · A. Cosar
Middle East University Computer Engineering Department, Universities Street,
6800 Ankara, Turkey
e-mail: tansel@ceng.metu.edu.tr

M. Beyaz
e-mail: muhammed.beyaz@ceng.metu.edu.tr

A. Cosar
e-mail: cosar@ceng.metu.edu.tr

O.H. Abdelrahman et al. (eds.), *Information Sciences and Systems 2015*,
Lecture Notes in Electrical Engineering 363,
DOI 10.1007/978-3-319-22635-4_19

objects (Orientation: whether the objects can be rotated or not is a key aspect of the 2DBPP) in packing creates better results but objects may not be rotatable in every problem definition. The textile industry can change the orientation of single color shirts by rotating the shirts while the process is in cutting phase, because there is no difference between rotation or not. However, the orientation of fragile items is important in shipping. If an item can be rotated then it is called as non-oriented or orientation-free. If an object of problem cannot be rotated it is called as oriented or orientation-fix. Online and offline are two categories of 2DBPP according to the availability of information about all items. Online bin packing (OnBP) means that objects arrive one by one and there is no way to know the complete input sequence, so it must be inserted into a bin immediately without waiting other items. Load balancing of 2DBPP, is the stabilization of total moments of rectangle items on the left of Centre of Gravity (CG) of bins with total moments of rectangles on the right of CG of bins. Euclidean Center of bin is considered as CG of a bin.

In the proposed algorithms, we use solution methods inspired from Evolutionary Algorithms (EA). Reproduction, mutation, recombination and selection are key mechanisms of EA that are used for solving NP-Hard optimization problems. BPP, Travelling Salesman and Quadratic Assignment Problem [2, 6] are well-known challenging NP-Hard problems modelled and solved successfully with EAs. Genetic Algorithm (GA) and Memetic Algorithm (MA) are the most efficient approaches of EAs. GA mimics the natural evolution process and has the ability to find (near-) optimal solutions in a large search space. Survival of the fittest individual is a rule allowing the best solution in each iteration to converge to a (near-) optimal solution in practical times. In GA, parents mate and produce offsprings and the best individuals are selected to survive to the next generation. MA is another growing area of EA. The fittest of individuals is selected as the solution of optimization problem. It mimics natural evolution process but it may differ from GA by performing individual learning which is also known as meme(s).

We propose two different Multiheuristic Multiobjective GA (MH-MOGA) that optimize both the minimal number and the load-balancing of the bins. The first proposed multiobjective algorithm, MHO-MOGA, uses heuristics: FNF, FFF, BFDH, UTS and LGFof to solve oriented multi objective 2D offline BPPs. The second proposed multiobjective algorithm, MHNO-MOGA, uses heuristics: FNF, FFF, BFDH, UTS, LGFof and LGFi and tries to find a pareto-optimal solution (minimal number of bins and most effective load-balancing of items) for non-oriented multiobjective 2D offline BPPs.

2 Formulation of the Load-Balancing Problem

Multiobjective 2DBPP with load balancing [7–9] tries to minimize Eq. 1

$$(C/2 + LB/2) \tag{1}$$

where

$$C = \sum_{j=1}^{n} c_j \tag{2}$$

$$LB = \sum_{j=1}^{C} \left| \sum_{i=1}^{B} p_{ij} d_i m_{ij} \sqrt{(x_{ij} + (w_{ij}/2) - x_{CG})^2 + (y_{ij} + (h_{ij}/2) - y_{CG})^2} \right| \tag{3}$$

which is subject to

$$x_i + (w_i w_i^x) + (h_i h_i^x) \leq x_k + (1 - le_{ik}), \quad \forall i, k, i < k \tag{4}$$

$$x_k + (w_k w_k^x) + (h_k h_k^x) \leq x_i + (1 - ri_{ik}), \quad \forall i, k, i < k \tag{5}$$

$$y_i + (w_i w_i^x) + (h_i h_i^x) \leq y_k + (1 - un_{ik}), \quad \forall i, k, i < k \tag{6}$$

$$y_k + (w_k w_k^x) + (h_k h_k^x) \leq y_i + (1 - ab_{ik}), \quad \forall i, k, i < k \tag{7}$$

$$le_{ik} + ri_{ik} + un_{ik} + ab_{ik} \leq p_{ij} + p_{kj} - 1, \quad \forall i, k, i < k \tag{8}$$

$$\sum_{j=1}^{C} p_{ij} = 1, \quad \forall i \tag{9}$$

$$\sum_{i=1}^{B} p_{ij} \leq M c_j, \quad \forall j \tag{10}$$

$$x_i + (w_i w_i^x) + (h_i h_i^x) \leq W_j + (1 - p_{ij}) M, \quad \forall i, j \tag{11}$$

$$y_i + (w_i w_i^y) + (h_i h_i^y) \leq H_j + (1 - p_{ij}) M, \quad \forall i, j \tag{12}$$

$$w_i^x, w_i^y, h_i^x, h_i^y, le_{ik}, ri_{ik}, ab_{ik}, un_{ik}, p_{ij}, c_j \in 0, 1, \quad \forall i, k, i < k \tag{13}$$

$$x_i, y_i \geq 0, \quad \forall i \tag{14}$$

$$m_{ij} \in -1, 1, \quad \forall i \tag{15}$$

$$x_{CG} \geq (W/2) \tag{16}$$

$$y_{CG} \geq (H/2) \tag{17}$$

B total number of rectangles
C total number of bins

LB	the total sum of load balancing
w_i, h_i	width and height of rectangle i
d_i	weight of rectangle i
W_j, H_j	width and height of bin j
x_i, y_i	left-bottom corner of rectangle i as coordinate
x_{CG}	x coordinate of center of gravity of bin which is equal to $(W/2)$
x_{CG}	y coordinate of center of gravity of bin which is equal to $(H/2)$
w_i^x, w_i^y	width of rectangle i is parallel to X and Y axis
h_i^x, h_i^y	height of rectangle i is parallel to X and Y axis
le_{ik}	rectangle i is placed on the left side of rectangle k
ri_{ik}	rectangle i is placed on the right side of rectangle k
ab_{ik}	rectangle i is placed above rectangle k
un_{ik}	rectangle i is placed under rectangle k
p_{ij}	$p_{ij} = 1$ if rectangle i is placed in bin j otherwise $p_{ij} = 0$
m_{ij}	$m_{ij} = 1$ if $(x_{ij} + (w_{ij}/2) - x_{CG}) \geq 0$ otherwise $m_{ij} = -1$
c_j	$c_j = 1$ if bin j is used otherwise $c_j = 0$
M	an arbitrarily large number used in Bin-M constraints

3 Proposed Algorithms

The chromosome is an array of values representing a possible solution to a 2DBPP. There are two parts in the chromosome. Rectangle items and a heuristics part that keeps the heuristics. Permutation encoding which is a form of keeping width-height of rectangular items and processing sequence between rectangles is used to keep the identification of rectangles. Gene (rectangle item) packing is done in two different ways. If Heu1 is equal to Heu2, then all genes are packed as a whole by using Heu1 and the result of Heu1 shows the number of required bins for solution. If Heu1 is different than Heu2, then Heu1 packs the first half of the genes and Heu2 packs the second half of the genes. The sum of required bins of Heu1 and Heu2 shows the number of required bins for solution. Elitist selection that gives higher chance to better chromosomes in the population is used in the proposed algorithms.

Single point crossover is used in our algorithms. Three different mutation operators are used in the algorithms in accordance with the orientation possibility of the items. The proposed mutation operators work on the rectangular items part of a chromosome. Swap mutation, rotation mutation (for non-oriented items), and swap-rotation mutation are the mutation operators.

The proposed MA mimics the natural evolution process. Our algorithms consider load balancing with center of gravity to each bin. In order to calculate center of gravity, each bin's center point is selected as Center of Gravity (CG). When a rectangle box is inserted, the Euclidean distance of its center to bins CG is calculated and multiplied by the weight of rectangle. This calculated value is CG of a rectangle. If sign of x coordinate of rectangle is minus then CG of rectangle is subtracted from total CG

of Bin. If sign of x coordinate of rectangle is plus then CG of rectangle is added to total CG of Bin. When all rectangles are inserted to bins, absolute values of total CG of bins are added. This calculated value is called as CG of a chromosome. The load balance of a bin is explained in Eq. 18 and load balance of a chromosome is explained in Eq. 19.

$$LB_{Bin} = \sum_{j=1}^{\#Rect} d_{ij} m_{ij} \sqrt{(x_{ij} + (w_{ij}/2) - x_{CG})^2 + (y_{ij} + (h_{ij}/2) - y_{CG})^2} \tag{18}$$

$$LB_{Chromosome} = \sum_{i=1}^{\#Bin} \left| \sum_{j=1}^{\#Rect} d_j m_{ij} \sqrt{(x_{ij} + (w_{ij}/2) - x_{CG})^2 + (y_{ij} + (y_{ij}/2) - y_{CG})^2} \right| \tag{19}$$

Multiheuristic Oriented Multiobjective GA (MHO-MOGA) is proposed to solve oriented multiobjective offline 2DBPP. FNF, FFF, BFDH, LGFof and UTS are applied as heuristics on the base of a GA. Swap mutation operation is used to keep orientation of items. Multiheuristic Non-Oriented Multiobjective GA (MHNO-MOGA) is developed for optimization of non-oriented items. Each individual uses one or two of the heuristics: FNF, FFF, BFDH, UTS, LGFof and LFGi to pack rectangles into bins. At the beginning of GA, each individual picks one of the heuristics. At the next phase, an individual can have two different heuristics. Each heuristic is applied to the corresponding part of rectangle list. Rotation mutation and swap-rotation mutation are used to change orientation of rectangles. Best result of GA becomes the solution of the problem.

4 Experimental Setup and Results

Two well known offline 2DBPP instance sets are used for the experiments (Berkey-Wang and Martello-Vigo) [10, 11]. Experimental setup consists of UTS, LFGi and GA whose heuristics are FNF, FFF, BFDH and LGFof. First, we apply GA to the problem and later the best result's rectangle list is given as input to UTS and LGFi. The parameter setting for the population size and the number of generations are decided to be 60 and 40 respectively. These parameter settings are used in all of the proposed algorithms throughout the experiments. The general results of MHNO-MOGA experiments are listed in Tables 1 and 2. We compared our results with LGFi algorithm.

In order to analyze runtime and efficiency of the algorithms, we randomly picked five different item size (20, 40, 60, 80, 100) 2DBPP. Each test is run for five times. Best values of FNF, FFF, BFDH, LGFof and LGFi are used as results. For the proposed algorithms and UTS, we used the average values of the experiments. Comparisons of algorithms according to 500 instance test setup are also explained in detail. The

Table 1 Result of MHNO-MOGA Ro for Berkey-Wang instances

Class	Num of rect	LGFi b	LGFi cg	Pr. algo b	Pr. algo cg
1	20	67	230.9	68	67.5
	40	131	392.6	131	218.7
	60	199	620.5	197	359.6
	80	271	877.5	270	584.6
	100	317	1123.7	320	676.6
	Av.	197	649	197.2	381.4
2	20	10	256.8	10	1.7
	40	20	530.6	20	6.4
	60	25	993.6	25	160.7
	80	32	1107.9	31	134.1
	100	39	1078.7	39	195.7
	Av.	25.2	793.5	25	99.7
3	20	50	918.7	49	241.3
	40	97	1706.4	95	643.1
	60	139	2785.7	139	1073
	80	189	3805.2	192	1709
	100	227	4422.1	229	2211.8
	Av.	140.4	2727.6	140.8	1175.6
4	20	10	1794.8	10	0.8
	40	19	3940.2	19	8
	60	26	4109.7	25	174.8
	80	33	5558.4	33	162.9
	100	39	7418.4	40	360.7
	Av.	25.4	4564.3	25.4	141.4
5	20	62	2097.4	60	555.3
	40	117	5039.1	117	2167.1
	60	180	7031	177	3292
	80	246	9068.5	242	4477
	100	290	11640	286	5573.6
	Av.	179	6975.2	176.4	3213
6	20	10	4788.4	10	8
	40	19	13578.4	19	14.6
	60	23	15069.6	22	202.6
	80	30	20176	30	999.9
	100	35	25989.9	34	1995.2
	Av.	23.4	15920.5	23	644.1

runtime analysis of FNF, FFF, BFDH, UTS, LGFof and MHO-MOGA for oriented multiobjective random picked tests are shown in Table 3. MHO-MOGA is reported to be the most time consuming algorithm.

Table 2 Result of MHNO-MOGA Ro for Martello-Vigo instances

Class	Num of rect	LGFi b	LGFi cg	Pr. algo b	Pr. algo cg
7	20	55	1648.6	53	1145.2
	40	109	3717.4	107	2968.7
	60	161	6041	155	3869.6
	80	223	6989.4	221	6229.4
	100	271	9330.1	264	7259.2
	Av.	163.8	5545.3	160	4294.4
8	20	56	1858.4	55	873.5
	40	111	3877.2	107	2052.2
	60	163	5440.7	155	3055.2
	80	221	7640.6	214	4818.7
	100	269	8981.9	264	6412.7
	Av.	164	5559.8	159	3442.5
9	20	143	3012.3	143	2060.1
	40	275	6081.2	275	4432.6
	60	435	10005.9	435	7661.3
	80	573	13963.7	573	9919.7
	100	693	16765.3	693	12324
	Av.	423.8	9965.7	423.8	7279.5
10	20	41	3800.9	43	396.2
	40	75	6710.7	75	732.4
	60	104	9447.3	105	1885.1
	80	133	12136.1	134	2937.9
	100	163	15835.2	165	4615.5
	Av.	103.2	9586	104.4	2113.4

Table 3 Runtime of algorithms for oriented multiobjective problems in msec

Rect	FNF	FFF	BFDH	UTS	LGFof	MHO-MOGA
20	4.1	4.6	4.7	81.7	64.0	17515
40	4.7	4.8	5.0	317.8	29.9	88024
60	5.3	6.4	7.8	27761.4	1529.8	3034228
80	7.6	8.4	9.2	3050.4	334.6	4664450
100	10.6	12.4	17.4	3746.9	291.1	677750
Av.	6.5	7.3	8.8	6991.6	449.9	1696393.4

The general results of second experiment are listed in Tables 4 and 5. We compared our results with LGFi algorithm.

Result (bin/cg) analysis of FNF, FFF, BFDH, UTS, LGFof and MHO-MOGA for oriented multiobjective random picked tests are shown in Table 6.

Table 4 Result of MHNO-MOGA SwRo for Berkey-Wang instances

Class	Num of rect	LGFi b	LGFi cg	Pr. algo b	Pr. algo cg
1	20	67	230.9	66	76.9
	40	131	392.6	125	202.8
	60	199	620.5	200	359.2
	80	271	877.5	264	630.7
	100	317	1123.7	320	711.7
	Av.	197	649	195	396.3
2	20	10	256.8	10	1
	40	20	530.6	20	8.7
	60	25	993.6	25	133
	80	32	1107.9	31	131.8
	100	39	1078.7	39	245.5
	Av.	25.2	793.5	25	104
3	20	50	918.7	49	260.3
	40	97	1706.4	97	581.1
	60	139	2785.7	141	1165.3
	80	189	3805.2	190	1775.7
	100	227	4422.1	228	2335.5
	Av.	140.4	2727.6	141	1223.6
4	20	10	1794.8	10	0.5
	40	19	3940.2	19	22.6
	60	26	4109.7	25	97.1
	80	33	5558.4	33	346.4
	100	39	7418.4	39	633.6
	Av.	25.4	4564.3	25.2	220
5	20	62	2097.4	60	719.4
	40	117	5039.1	116	2027
	60	180	7031	177	3444.2
	80	246	9068.5	245	5043.8
	100	290	11640	286	5154.2
	Av.	179	6975.2	176.8	3277.7
6	20	10	4788.4	10	4.9
	40	19	13578.4	19	51.3
	60	23	15069.6	22	112.3
	80	30	20176	30	502
	100	35	25989.9	34	1356.8
	Av.	23.4	15920.5	23	405.5

Table 5 Result of MHNO-MOGA SwRo for Martello-Vigo instances

Class	Num of rect	LGFi b	LGFi cg	Pr. algo b	Pr. algo cg
7	20	55	1648.6	53	1015.6
	40	109	3717.4	108	2612.5
	60	161	6041	156	3880.6
	80	223	6989.4	220	6126.4
	100	271	9330.1	267	6899.4
	Av.	163.8	5545.3	160.8	4106.9
8	20	56	1858.4	53	785.3
	40	111	3877.2	107	2170.4
	60	163	5440.7	155	3044
	80	221	7640.6	214	4521.9
	100	269	8981.9	265	6237.6
	Av.	164	5559.8	158.8	3351.8
9	20	143	3012.3	143	1885.9
	40	275	6081.2	275	4241.6
	60	435	10005.9	435	6762.7
	80	573	13963.7	573	9355.1
	100	693	16765.3	693	11598.4
	Av.	423.8	9965.7	423.8	6768.7
10	20	41	3800.9	42	263.2
	40	75	6710.7	75	971
	60	104	9447.3	102	2616.8
	80	133	12136.1	133	3837.2
	100	163	15835.2	166	4662.4
	Av.	103.2	9586	103.6	2470.1

Results (bin/cg) of FNF, FFF, BFDH, UTS, LGFof and MHO-MOGA for oriented single objective 500 problem set are shown in Table 7.

Superiority of MHO-MOGA to FNF, FFF, BFDH, UTS and LGFof for oriented multiobjective 500 problem set are shown in Table 8.

Runtime analysis of FNF, FFF, BFDH, UTS, LGFof, LGFi and MHNO-MOGA for non-oriented multiobjective random picked tests are shown in Table 9. MHNO-MOGA is reported to be the most time consuming algorithm.

Result (bin/cg) analysis of FNF, FFF, BFDH, UTS, LGFof, LGFi and MHNO-MOGA for non-oriented multiobjective random picked tests are shown in Table 10.

Results (bin/cg) of FNF, FFF, BFDH, UTS, LGFof, LGFi and MHNO-MOGA (r) for non-oriented multiobjective 500 problem set (according to continuous lower bound) are shown in Table 11.

Superiority of MHO-MOGA (r) versus FNF, FFF, BFDH, UTS, LGFof and LGFi for non-oriented multiobjective 500 problem set are shown in Table 12.

Table 6 Results (bin/cg) of algorithms for oriented multiobjective problems

Rect	FNF	FFF	BFDH	UTS	LGFof	MHO-MOGA
20	11	9	9	9	9	9
	310.4	167.3	151.3	491.1	301.4	116.6
40	17	13	13	13	12	12
	62.4	59.8	50	64.5	33.2	19.6
60	53	47	47	47	47	46
	1207.5	1024.7	1008.5	1103.9	1021.1	813.7
80	30	24	23	24	24	24
	1704.5	1850.1	1440.7	900.9	1728.7	421
100	44	31	31	30	30	29
	1623.9	1391.2	1098.2	1088.9	1331.2	769.9
Av.	31	24.8	24.6	24.6	24.6	24
	981.5	898.6	749.7	729.9	883.1	428.2

Table 7 Results (bin/cg) of heuristics and MHO-MOGA for oriented multiobjective 500 problem set

Total	FNF	FFF	BFDH	UTS	LGFof	Pr. alg.
Bin	9489	7591	7514	7521	7430	7389
CG	340,954	333,247	319,187	297,483	347,076	126,581

Table 8 MHO-MOGA versus heuristics for oriented multiobjective 500 problem set

	FNF	FFF	BFDH	UTS	LGFof
MHO-MOGA (%)	100	100	100	100	97.2

Table 9 Runtime of algorithms for non-oriented multiobjective problems in msec

Rect	FNF	FFF	BFDH	UTS	LGFof	LGFi	MHNO-MOGA
20	4.1	4.6	4.7	81.7	64.0	65.0	87,615
40	4.7	4.8	5.0	317.8	29.9	30.1	39,027
60	5.3	6.4	7.8	27761.4	1529.8	2416.4	1545,971
80	7.6	8.4	9.2	3050.4	334.6	257.4	645,442
100	10.6	12.4	17.4	3746.9	291.1	324.3	257,878
Av.	6.5	7.3	8.8	6991.6	449.9	618.6	515186.6

Table 10 Results (bin/cg) of algorithms for non-oriented multiobjective problems

Rect	FNF	FFF	BFDH	UTS	LGFof	LGFi	MHNO-MOGA
20	11	9	9	9	9	8	8
	310.4	167.3	151.3	491.1	301.4	150.9	60
40	17	13	13	13	12	12	12
	62.4	59.8	50	64.5	33.2	51.9	15
60	53	47	47	47	47	46	46
	1207.5	1024.7	1008.5	1103.9	1021.1	901.4	700.2
80	30	24	23	24	24	22	21
	1704.5	1850.1	1440.7	900.9	1728.7	887.4	821.3
100	44	31	31	30	30	29	28
	1623.9	1391.2	1098.2	1088.9	1331.2	1131.2	575
Av.	31	24.8	24.6	24.6	24.6	23.4	23
	981.5	898.6	749.7	729.9	883.1	624.6	434.3

Table 11 Results (bin/cg) of heuristics and MHNO-MOGA (r) for non-oriented multiobjective 500 problem set

Total	FNF	FFF	BFDH	UTS	LGFof	LGFi	Pr. Alg.
Bin	9489	7591	7514	7521	7430	7226	7175
CG	340,954	333,247	319,187	297,483	347,076	311,434	113,925

Table 12 MHO-MOGA(r) versus heuristics for non-oriented multiobjective 500 problem set

	FNF	FFF	BFDH	UTS	LGFof	LGFi
MHNO-MOGA(r) (%)	100	100	100	100	100	95

Table 13 Results (bin/cg) of heuristics and MHNO-MOGA(sr) for non-oriented multiobjective 500 problem set

Total	FNF	FFF	BFDH	UTS	LGFof	LGFi	Pr. Alg.
Bin	9489	7591	7514	7521	7430	7226	7165
CG	340,954	333,247	319,187	297,483	347,076	311,434	111,623

Results (bin/cg) of FNF, FFF, BFDH, UTS, LGFof, LGFi and MHNO-MOGA (sr) for non-oriented multiobjective 500 problem set (according to continuous lower bound) are shown in Tables 11 and 13.

Superiority of MHNO-MOGA (sr) versus FNF, FFF, BFDH, UTS, LGFof and LGFi for non-oriented multiobjective 500 problem set are shown in Table 14.

Table 14 MHO-MOGA (sr) versus heuristics for non-oriented multiobjective 500 problem set

	FNF	FFF	BFDH	UTS	LGFof	LGFi
MHNO-MOGA(sr) (%)	100	100	100	100	100	96

5 Conclusions and Future Work

In this study, two novel robust algorithms for the multiobjective optimization of offline 2DBPP are proposed. Our experimental results show that while the well known heuristics sometimes produce better results for minimizing the number of bins in 2DBPP they are not efficient for solving the multiobjective load balancing problem of 2D bins while also minimizing bins. MHO-MOGA and MHNO-MOGA give better results not only for minimum number of bins but also for the load balancing of 2D bins. MHNO-MOGA makes use of rotation and swap-rotation mutation operators. MHNO-MOGA with rotation mutation outperforms LGFi heuristic for 95.0 % of the problems. MHNO-MOGA with swap-rotation mutation outperforms LGFi heuristic for 96.0 % of the problems and 97.2 % of the benchmark problems for the LGFof heuristic.

References

1. Johnson, D.S.: Near-optimal bin-packing algorithms. Ph.D. thesis (1973)
2. Dokeroglu, T., Cosar, A.: Optimization of one-dimensional bin packing problem with island parallel grouping genetic algorithms. Comput. Ind. Eng. **75**, 176–186 (2014)
3. Coffman, E.G., Shor, P.W.: Average-case analysis of cutting and packing in two dimensions. Eur. J. Oper. Res. **44**, 134–144 (1990)
4. Dokeroglu, T.: Hybrid teaching-learning-based optimization algorithms for the Quadratic Assignment Problem. Comput. Ind. Eng. **85**, 86–101 (2015)
5. Lodi, A., Martello, S., Monaci, M.: Two-dimensional packing problems: A survey. Eur. J. Oper. Res. **141**(2), 241–252 (2002)
6. Tosun, U., Dokeroglu, T., Cosar, A.: A robust island parallel genetic algorithm for the quadratic assignment problem. Int. J. Prod. Res. **51**(14), 4117–4133 (2013)
7. Blum, C., Schmid, V.: Solving the 2d bin packing problem by means of a hybrid evolutionary algorithm. Procedia Comput. Sci. **18**(0), 899–908 (2013). International Conference on Computational Science (2013)
8. Fernandez, A., Gil, C., Banos, R., Montoya, M.G.: A parallel multiobjective algorithm for two-dimensional bin packing with rotations and load balancing. Expert Syst. Appl. **40**(13), 5169–5180 (2013)
9. Thapatsuwan, P., et al.: Development of a stochastic optimisation tool for solving the multiple container packing problems. Int. J. Prod. Econ. **140**(2), 737–748 (2012)
10. Smith, J.E.: Coevolving memetic algorithms: A review and progress report. Syst., Man, Cybern., Part B: Cybern. **37**(1), 6–17 (2007)
11. Berkey, J.O., Wang, P.Y.: Two-dimensional finite bin-packing algorithms. J. Oper. Res. Soc. **38**, 423–429 (1987)