A Genetic Algorithm for Two-Dimensional Bin Packing Problem

Lee Lai Soon

Laboratory of Applied and Computational Statistics
Institute of Mathematical Research
Universiti Putra Malaysia

Department of Mathematics
Faculty of Science
Universiti Putra Malaysia

lee@math.upm.edu.my

Abstract

This paper considers a non-oriented two-dimensional bin packing problem, where a set of small rectangles, which may be rotated by 90°, has to be allocated to one or more identical bins. A genetic algorithm (GA) hybridised with a heuristic placement routine is proposed to solve the problem. The algorithm employs an innovative crossover operator that considers a number of different children from each pair of parents. Comprehensive results are presented, and the algorithm is shown to be competitive when compared with other metaheuristic algorithms.

Introduction

Cutting and Packing problems are optimization problems that are concerned with finding a good arrangement of multiple small items in one or more larger objects. This type of problem is encountered in many areas of industry such as wood, glass, and textile industries, newspaper paging, cargo loading, and etc. The usual objective is aimed at maximising the utilisation of the large objects, or maximising the value of the small items packed. This work is focused on the non-oriented case of the two-dimensional rectangular bin packing problem with the objective of packing without overlaps, all the rectangles into the minimum number of bins.

A considerable amount of research has been carried out and various heuristic and metaheuristic approaches have been proposed to solve the problem. Some excellent and comprehensive reviews of the approaches can be found in Dowsland and Dowsland [5], Hopper and Turton [6], and Lee [7].

This paper focuses on the development of a new strain of GA called MultiCrossover Genetic Algorithm (MXGA) in comparison with other metaheuristic algorithms namely Standard Genetic Algorithm (SGA), Unified Tabu Search (UTS) and Randomised Descent Method (RDM) for solving the problem. A common feature found in the metaheuristic algorithms developed is their two-stage approach, where the algorithms are combined with a heuristic placement routine. In this two-stage approach, an algorithm manipulates the encoded solutions, which are then evaluated by the placement routine transforming the packing sequence into the corresponding physical layout.

Heuristic placement routine

Inspired by the best-fit heuristic proposed by Burke *et al.* [3] and Whitwell [10] for solving strip packing problem, Bennell *et al.* [1] developed a heuristic placement routine called Lowest Gap Fill (LGF) that is effective in filling the available gaps in the partial layout by dynamically selecting the best rectangle for placement during the packing stage.

LGF consists of two stages: *preprocessing stage* and *packing stage*. As in the best-fit heuristic, before the start of the LGF placement routine, the rectangles are initially arranged following a horizontal orientation and sorted in a non-increasing order of their width, breaking ties by non-increasing height. This preprocessing stage required $O(n \log n)$ time, where n is the number of rectangles.

LGF employs a best-fit type strategy by examining the lowest available gap in the current bin and then placing the rectangle that best fits the gap available. This routine not only keeps track of the free position in the layout, but also of the dimensions of the available gap at the respective position. When no remaining rectangle can fit into any of the available gaps in the current bin, the bin is closed and a new empty bin is initialised to replace the closed bin as the current bin. Any unfilled space in the closed bin will be regarded as wastage. The routine continues until all the rectangles in the list have been packed into a minimum number of bins. This packing stage requires $O(n^2)$ time. The LGF

is sufficiently fast to be used repeatedly within a metaheuristic algorithm, and is comparable in performance with existing heuristics that are computationally more expensive.

Metaheuristic algorithms

MultiCrossover genetic algorithm (MXGA)

A key feature of the MXGA is the application of a simple yet effective crossover operator for multiple times to each set of parents in order to optimise the quality of the child chromosomes. A more detailed description of the MXGA is given by Lee [7].

In the MXGA, the complete set of rectangles n, forms the length of the chromosome (individual). The genes are represented by a uniform random permutation of the integer numbers of bins in the interval $[1, LB]$, where LB is the lower bound of Dell'Amico *et al.* [4]. Thus, a solution to the packing problem in this case consists of a sequence of positive integer numbers indicating the bin number, in which the rectangles are placed into the bin. The exact location in the layout is then determined by the heuristic placement routine.

The probabilistic binary tournament selection scheme is used as the selection mechanism of each parent in the MXGA. Two individuals are chosen at random, and then a random number r is generated from the interval $[0,1]$. If $r < s$, where s is a parameter, the fitter of the two individuals is selected to be the parent; otherwise, the less fit individual is selected. The two individuals are then returned to the original population and can be selected again.

The multicrossover operator in MXGA has the 1-point crossover strategy at its core. Instead of performing a single 1-point crossover with each pair of parents, the crossover process is repeated t times to produce $2t$ valid temporary offspring. Then, the best and a selected valid temporary offspring (using the probabilistic binary tournament selection) are chosen to be the offspring for the current generation.

Note that the crossover operator is only applied to the selected parents with a given crossover probability, p_c. When crossover is not applied, a swap operator is used to generate new offspring that is different from their parent. The basic step of this operator is to randomly select a swap point in a parent, and then swap the substrings separated by the swap point to form an offspring.

The mutation operator is applied in two stages. First, a subset of individuals is selected from the new offspring population with a given individual mutation probability p_M. Then each gene in the selected offspring is considered in turn, and the bin number is randomly changed to an element of the set $\{1, 2, \cdots, LB\}$ according to the gene mutation probability p_m.

The elitism replacement scheme is used whereby the offspring have to compete with their parents to gain admission to the new population. After the new population has been selected, a process called *filtration* is used to identify the identical individuals from the new population. Any duplicate individuals are removed and replaced by uniformly randomly generated new individuals to avoid premature convergence and to add diversity to the new population. Because of the computational cost of the filtration procedure, this procedure is invoked every R generations.

Standard genetic algorithm (SGA)

The SGA differs from the MXGA in the use of the crossover operator, reproduction procedure and the replacement scheme. The SGA applies the 1-point crossover operator only once to produce two offspring from two selected parents, and uses the reproduction procedure instead of a swap operator. Hence, when the crossover does not apply to the selected parents, they are simply reproduced for the next generation. Finally, the replacement strategy is steady-state.

Unified tabu search (UTS)

The main feature of the UTS framework is the use of a unified parametric neighbourhood, whose size and structure are dynamically varied during the search. The algorithm also adopts a search scheme which is independent of the specific packing problem to be solved. More precisely, at each iteration of the main loop, the size and the structure of a neighbourhood are determined and a specific procedure is invoked to explore it. Based on the output, the exploration is iterated until the stopping criterion is met. Detailed description of the algorithm can be found in Lodi *et al.* [8].

Given a current solution, the neighbourhood is searched through moves which consist of modifying the solution by changing the packing of a subset of rectangle S, in an attempt to empty a specific target bin which is selected based on a filling function. Each neighbourhood has a tabu list and a tabu tenure and is defined by the removal of a rectangle j contained in the target bin, and the repacking of j together with contents of k other bins. The value of k which defines the size of the

neighbourhood is varied dynamically as the algorithm progresses.

Randomised descent method (RDM)

The RDM uses an adaptation of the UTS. The main difference lies in the removal of the tabu list and an alternative acceptance rule. The acceptance rule in RDM allows the neutral moves for up to 1000 consecutive iterations before terminating the algorithm. When there are multiple identical neutral moves found during the neighbourhood search procedure in a single iteration, randomization is used to randomly select a move from the list of moves with the same objective function value. Consequently, the procedure can escape from falling into the same local optimum and continue its search. Note that deteriorating moves are not considered in RDM.

Experimental design

The four metaheuristic algorithms are coded in ANSI-C using Microsoft Visual C++ 6.0 as the compiler. Ten different classes of problem instances that have formed the basis for comparing algorithms in previous studies reported in the literature are used. These classes are listed in Table 1. The first six classes (I - VI) are introduced by Berkey and Wang [2], while the other four classes (VII - X) are introduced by Martello and Vigo [9] and are based on the following types of rectangles:

Type 1: w_j uniformly random in $\left[\frac{2}{3}W, W\right]$;
h_j uniformly random in $\left[1, \frac{1}{2}H\right]$.

Type 2: w_j uniformly random in $\left[1, \frac{1}{2}W\right]$;
h_j uniformly random in $\left[\frac{2}{3}H, H\right]$.

Type 3: w_j uniformly random in $\left[\frac{1}{2}W, W\right]$;
h_j uniformly random in $\left[\frac{1}{2}H, H\right]$.

Type 4: w_j uniformly random in $\left[1, \frac{1}{2}W\right]$;
h_j uniformly random in $\left[1, \frac{1}{2}H\right]$.

For each class, five values of n = 20; 40; 60; 80; 100 are considered. Also, for each combination of class and value of n, 10 problem instances are generated.

The problem instances are provided by Lodi *et al.* [8] and are publicly available at http://www.or.deis.unibo.it/research.html.

The specific values for the generic design variables in MXGA and SGA are summarised in Table 2.

Since the optimal solutions for the problem instances are not known, the lower bound proposed by Dell'Amico *et al.* [4] is used. The performance of the various metaheuristic algorithms and the heuristic placement routine is compared on the basis of the Ratio and the Overall Bin Utilisation (OBU) defined by:

$$\text{Ratio} = \frac{UB_i}{LB_i} \qquad (1)$$

$$\text{OBU} = \sum_{j=1}^{UB_i} \frac{A_j}{W \times H} \qquad (2)$$

Note that the variables UB_i and LB_i represent the heuristic solutions found and the lower bound for the problem instance i. A_j in equation (2) is the total area of rectangles in bin j $\left(j = 1, 2, \cdots, UB_i\right)$ for problem instance i.

Results and discussion

The computational results of the metaheuristic algorithms and the heuristic placement routine are presented in Table 3. For each algorithm, the entries in the first column report the average Ratio, while the entries in the second column give the average percentage OBU, both computed over the ten problem instances for each class and each value of n. The final line for each data class gives the average overall values over that class, and the final line of the table gives the overall average value over all classes. The values in bold represent the best solution found in each class. For a fair comparison between the different metaheuristic algorithms, a stopping criterion of 120 CPU seconds per problem instance is employed. Also note that the execution time for the heuristic placement routine is less than 0.1 CPU second per problem instance.

Table 1: Classes for the problem instances

Class	Bin $(W \times H)$	Item $(w_j$ and $h_j)$
I	10×10	uniformly random in $[1,10]$
II	30×30	uniformly random in $[1,10]$
III	40×40	uniformly random in $[1,35]$
IV	100×100	uniformly random in $[1,35]$
V	100×100	uniformly random in $[1,100]$
VI	300×300	uniformly random in $[1,100]$
VII	100×100	Type 1 with probability 70% Type 2, 3, 4 with probability 10% each
VIII	100×100	Type 2 with probability 70% Type 1, 3, 4 with probability 10% each
IX	100×100	Type 3 with probability 70% Type 1, 2, 4 with probability 10% each
X	100×100	Type 4 with probability 70% Type 1, 2, 3 with probability 10% each

Table 2: Implementation of generic design variables for MXGA and SGA

variable	value
chromosome length, L	n (number of rectangles)
population size, P_{pop}	100
crossover probability, p_c	0.75
multicrossover rate, t (MXGA only)	5
individual mutation probability, p_M	0.25
gene mutation probability, p_m	$1/n$
filtration rate, R (MXGA only)	every 50 generation

By considering for each class, the average values computed over all values of n, it is clear that the MXGA is producing the equal best or better results for all but class IV and VI with respect to average Ratio, and all but class II, IV and X with respect to average OBU.

In general, SGA produced the least impressive results. Only a small fraction of improvement (i.e. 0.5%) in term of the average Ratio is achieved in the SGA compared to the LGF placement routine. This may suggest that either the SGA is not an ideal choice of algorithm to be used in the bin packing problem or the LGF placement routine is itself already a powerful heuristic for the packing problem.

A closer scrutiny of the results for UTS and RDM show that both algorithms exhibit reasonable performance, with UTS performing marginally better. This supports the idea that the acceptance rule and randomization procedure introduced into RDM are comparable with the ideas of tabu lists and tabu tenure used in the UTS in generating high-quality solutions. This indicates that the randomisation procedure is capable of directing the moves to escape from local optima.

Improvements of 2.5% in the MXGA compared to LGF placement routine show that the MXGA is able to produce better solution quality. A significant improvement of 2% in the MXGA compared to SGA is obtained from the overall results. This suggests that the various techniques used in the MXGA are capable of improving the results although fewer generations are generated within the time limit. The overall results show that the MXGA algorithm is the preferred choice followed by the UTS, RDM and SGA.

Table 3: Comparison of LGF with MXGA, SGA, UTS and RDM

Class	n	LGF		MXGA		SGA		UTS		RDM	
		Ratio	OBU	Ratio	OBU	Ratio	OBU	Ratio	OBU	Ratio	OBU
I	20	1.03	80.54	1.03	81.48	1.03	81.11	1.03	81.31	1.05	80.69
	40	1.04	84.95	1.04	85.94	1.04	85.79	1.04	85.75	1.05	84.36
	60	1.05	86.56	1.04	88.40	1.05	86.63	1.04	88.05	1.04	87.73
	80	1.06	86.22	1.06	87.40	1.06	87.00	1.06	87.25	1.06	87.23
	100	1.04	91.56	1.02	93.43	1.03	92.26	1.03	92.45	1.03	92.67
Average		1.044	85.97	**1.037**	**87.33**	1.041	86.56	1.039	86.96	1.046	86.54
II	20	1.00	42.40	1.00	42.40	1.00	42.40	1.00	42.40	1.00	42.40
	40	1.10	54.72	1.10	56.07	1.10	54.80	1.10	56.07	1.10	56.07
	60	1.05	74.68	1.00	76.54	1.20	67.24	1.00	76.81	1.00	76.81
	80	1.07	78.48	1.00	83.51	1.07	78.00	1.00	83.45	1.03	81.34
	100	1.03	78.09	1.00	81.48	1.03	78.21	1.00	81.48	1.03	78.09
Average		1.050	65.67	**1.020**	68.00	1.080	64.13	**1.020**	**68.04**	1.032	66.94
III	20	1.06	65.70	1.04	68.91	1.06	66.18	1.04	68.91	1.06	66.02
	40	1.13	69.26	1.09	74.33	1.09	73.20	1.10	72.87	1.09	74.11
	60	1.10	77.12	1.08	81.76	1.10	77.71	1.09	80.23	1.09	80.25
	80	1.10	78.47	1.06	83.33	1.09	79.25	1.09	79.82	1.09	79.10
	100	1.08	81.05	1.06	85.27	1.08	81.10	1.06	83.34	1.07	83.01
Average		1.093	74.32	**1.065**	**78.72**	1.085	75.49	1.077	77.03	1.080	76.50
IV	20	1.00	38.36	1.00	38.36	1.00	38.36	1.00	38.36	1.00	38.36
	40	1.00	56.71	1.00	56.71	1.00	55.07	1.00	56.71	1.00	56.71
	60	1.15	67.58	1.10	71.46	1.10	70.07	1.10	71.05	1.10	71.07
	80	1.10	74.13	1.07	76.49	1.10	72.96	1.03	79.25	1.07	76.24
	100	1.07	75.25	1.03	79.30	1.07	75.75	1.03	79.15	1.03	79.15
Average		1.063	62.41	1.040	64.47	1.053	62.44	**1.033**	**64.90**	1.040	64.31
V	20	1.09	65.12	1.04	70.58	1.06	68.03	1.06	68.34	1.04	70.44
	40	1.10	71.58	1.06	76.54	1.08	73.74	1.09	72.89	1.07	75.21
	60	1.09	75.26	1.07	78.23	1.09	75.37	1.09	75.46	1.07	77.65
	80	1.09	75.12	1.07	78.96	1.08	76.35	1.08	76.63	1.07	78.85
	100	1.08	79.11	1.05	83.93	1.07	80.98	1.08	79.44	1.07	81.89
Average		1.092	72.24	**1.058**	**77.65**	1.076	74.89	1.079	74.55	1.064	76.81
VI	20	1.00	29.23	1.00	29.23	1.00	29.23	1.00	29.23	1.00	29.23
	40	1.40	47.55	1.40	49.09	1.40	47.40	1.30	50.12	1.40	48.25
	60	1.05	66.35	1.00	70.03	1.05	66.22	1.05	66.00	1.03	68.34
	80	1.00	68.67	1.00	68.67	1.00	66.66	1.00	68.67	1.00	68.67
	100	1.07	75.21	1.07	75.99	1.10	72.60	1.07	75.34	1.07	75.25
Average		1.103	57.40	1.093	**58.60**	1.110	56.42	**1.083**	57.87	1.100	57.95
VII	20	1.19	65.34	1.11	71.94	1.13	68.96	1.13	68.51	1.13	68.61
	40	1.12	75.74	1.07	80.43	1.09	77.16	1.08	78.97	1.08	78.89
	60	1.10	78.32	1.05	85.22	1.08	80.60	1.07	82.74	1.06	83.24
	80	1.10	81.05	1.08	84.79	1.10	81.23	1.10	81.14	1.10	81.33
	100	1.09	83.54	1.07	86.30	1.10	82.00	1.08	84.36	1.08	84.11
Average		1.119	76.80	**1.075**	**81.74**	1.101	77.99	1.093	79.14	1.090	79.24
VIII	20	1.15	66.54	1.10	72.14	1.12	69.27	1.10	72.14	1.10	72.14
	40	1.16	71.85	1.09	80.23	1.11	76.55	1.13	74.95	1.11	76.25
	60	1.09	81.36	1.06	84.93	1.10	79.96	1.07	83.27	1.07	83.01
	80	1.10	81.11	1.07	85.21	1.10	81.22	1.09	82.63	1.10	81.11
	100	1.09	82.12	1.06	86.55	1.09	82.31	1.08	84.74	1.08	84.56
Average		1.116	76.60	**1.078**	**81.81**	1.105	77.86	1.094	79.55	1.092	79.41
IX	20	1.01	43.03	1.00	43.57	1.01	43.03	1.00	43.57	1.00	43.57
	40	1.02	44.78	1.01	45.75	1.01	45.64	1.01	45.73	1.01	45.75
	60	1.01	43.44	1.01	43.56	1.01	43.47	1.01	43.56	1.01	43.56
	80	1.01	45.03	1.01	45.12	1.01	45.03	1.01	45.03	1.01	45.12
	100	1.01	46.10	1.01	46.10	1.01	45.99	1.01	46.10	1.01	46.10
Average		1.011	44.48	**1.007**	**44.82**	1.008	44.63	**1.007**	**44.82**	**1.007**	**44.82**
X	20	1.20	62.25	1.13	68.40	1.13	67.13	1.15	66.34	1.13	67.01
	40	1.07	77.45	1.06	79.87	1.06	78.17	1.06	79.73	1.06	79.55
	60	1.08	82.66	1.07	84.42	1.10	79.99	1.06	85.21	1.07	84.67
	80	1.06	85.25	1.06	85.83	1.07	82.66	1.05	89.14	1.05	89.00
	100	1.07	82.45	1.04	87.85	1.07	82.80	1.04	86.89	1.05	85.65
Average		1.098	78.01	**1.070**	81.27	1.086	78.15	1.072	**81.46**	1.072	81.18
AVERAGE		1.079	69.39	**1.054**	**72.44**	1.074	69.86	1.060	71.43	1.062	71.36

References

[1] Bennell, J.A., Lee, L.S. and Potts, C.N. (submitted). A Genetic Algorithm for Two-Dimensional Bin Packing with Due Dates. *INFORMS Journal on Computing.*

[2] Berkey, J.O. and Wang, P.Y. 1987. Two-Dimensional Finite Bin-Packing Algorithms. *The Journal of The Operational Research Society*, **38**(5): 423 – 429.

[3] Burke, E.K., Kendall, G. and Whitwell, G. 2004. A New Placement Heuristic for the Orthogonal Stock-Cutting Problem. *Operations Research*, **52**(4): 655 – 671.

[4] Dell'Amico, M., Martello, S. and Vigo, D. 2002. A Lower Bound for the Non-Oriented Two-Dimensional Bin Packing Problem. *Discrete Applied Mathematics*, **118**: 13 – 24.

[5] Dowsland, K.A. and Dowsland, W.B. 1992. Packing Problems. *European Journal of Operational Research*, **56**: 2 – 14.

[6] Hopper, E. and Turton, B.C.H. 2001. A Review of the Application of Meta-Heuristic Algorithms to 2D Strip Packing Problems. *Artificial Intelligence Review,* **16**: 257–300.

[7] Lee, L.S. 2006. *MultiCrossover Genetic Algorithms for Combinatorial Optimisation Problems.* Ph.D. Thesis. School of Mathematics, University of Southampton, United Kingdom.

[8] Lodi, A., Martello, S. and Vigo, D. 1999. Heuristic and Metaheuristic Approaches for a Class of Two-Dimensional Bin Packing Problems. *INFORMS Journal on Computing*, **11**: 345 – 357.

[9] Martello, S. and Vigo, D. 1998. Exact Solution of the Two-Dimensional Finite Bin Packing Problem. *Management Science*, **44**(3): 388 – 399.

[10] Whitwell, G. 2004. *Novel Heuristic and Metaheuristic Approaches to Cutting and Packing.* PhD Thesis. School of Computer Science and Information Technology, University of Nottingham, United Kingdom.