Design of Multi-dimensional Bin-packing Heuristic Based on Genetic Algorithm

Ping Zhang[a], Ju Jiang[a], Haiyan Xu[b], Xueshan Han[b]

[a]College of Automation Engineering, [b]College of Economics and Management

Nanjing University of Aeronautics and Astronautics

zhangping_nuaa@hotmail.com

DOI : 10.5963/IJCSAI0101002

***Abstract*-Genetic algorithm (GA) is an intelligent method based on natural selection for global numerical optimization. To make use of the great advantages of GA, this paper focuses on applying GA to multi-dimensional bin-packing problem (MBP). In multi-dimensional bin-packing problem, the primary goal is putting grouping items into appropriate bins with the same dimensions to make full use of the bins and minimize the number of bins used. In other words, the optimal solution requires that most bins are filled in maximum degree. By designing proper fitness functions, GA is successfully applied to MBP. The experimental results reveal that the proposed GA method solves the MBP efficiently.**

Keywords-Genetic Algorithm; Multi-dimensional Bin-packing; Fitness Function

I. INTRODUCTION

During the past decade, bin-packing (BP) problem [1, 2, 3] has become a research hotspot in combinatorial optimization. A lot of researchers have devoted to the theoretical developments, thus many heuristic algorithms are explored for the bin-packing problem. However, the topics are far from ending, there are no mature theory and exact method up to now. Under this circumstance, researchers mainly use approximate heuristic algorithm to deal with BP problem. Johnson analyzed the performance of the First-Fit (FF), Best-Fit (BF), First-Fit Decreasing (FFD) and Best-Fit Decreasing (BFD) heuristics in [2]. Bhatia proposed a Better-Fit Heuristic Method for One-dimension BP Problem in [4]. In recent years, more work has been done to develop the algorithms for two-dimension and three-dimension BP problem [5, 6]. Besides that, intelligent methods are applied to BP problem to optimal the above methods, such as, ACO, PSO.

Taking the process of rotation in real packing situation into consideration, Genetic algorithm (GA) is the most common used one in MBP which can simulate the process of rotation by designing suitable chromosome and fitness function. GA [7] is an efficient method for global optimization. So, it is suitable to generalize GA method to BP problem. There are many applications of GA for BP problem. A Grouping Genetic Algorithm was developed by Falkenauer in [6]. And in [8], Falkenauer developed a new representation and operators of GA for BP problem.

In this paper, due to the sort factor in BFD is special, an improved BFD (IBFD) will be introduced. The aim of this paper is to develop an improved heuristic algorithm based on GA (MBPGA) which will be introduced in details later. This algorithm discusses the way of solving the MBP with dimensions bigger than 3 (d>3). The goal of this algorithm is to improve IBFD to seek the optimal solution that the sum of every dimension of the grouping items in each bin is less than or equal to the capacity of bin. In this paper, the predefined bins and grouping items are all rotatable.

The rest of the paper is organized as follows. Some existed heuristic algorithms together with definitions for MBP (d>3) are presented in Section 2. Then, a comparison is presented between IBFD method and MBPGA which lead out the way defining fitness functions for the MBP in Section 3. Within Section 4, a practical application for MBP (d>3) is given. Finally, conclusions and future work are provided in Section 5.

II. HEURISTIC ALGORITHM AND DEFINITIONS FOR MBP

In this section, some existed heuristic algorithms for BP problem will be introduced and some definitions for MBP are stated.

A. Some Popular Heuristic Algorithm

Various heuristic on-line algorithms as well as off-line methods have been applied to BP problem by now [9]. Generally, on-line algorithms pack the items to a bin in sequence of arrival while off-line methods have all items available before packing. Three on-line (NF, FF, BF) together with two off-line methods are introduced as follows.

1) On-line algorithms

- Next-Fit (NF): NF keeps a bin active permanently, it puts the item into the bin if it fits in the bin; otherwise the heuristic uses a new bin for the item which is marked as the active bin.
- First Fit (FF): FF checks all the non-empty bins and packs the item into the first bin it fits; otherwise FF packs the item in a new bin.
- Best Fit (BF): BF packs the item into a bin that is filled to maximum degree but still having enough vacant space to pack the item; otherwise BF packs the item in a new bin.

2) Off-line algorithms

- First-Fit Decreasing (FFD): FFD packs the items according to the FF after sorting the list of items in non-increasing order of sizes.
- Best-Fit Decreasing (BFD): BFD packs the items according to the BF after sorting the list of items in non-increasing order of sizes.

Among the above methods, BFD is the best one with most efficiently packing effect. This is due to its sorting method to packing the biggest item into the smallest bin which has enough vacant space, which makes it possible to achieve a goal of making the most use of available bins.

B. Concepts and Definitions for MBP

In this part, some definitions and the Improved BFD (IBFD) are introduced.

Definition 1

A sign function, sign(x), is defined by

$$sign(x) = \begin{cases} 1 \ , x > 0 \\ 0 \ , x = 0 \\ -1, x < 0 \end{cases} \tag{1}$$

Definition 2

Let scalar d be the dimensions of the items and bins, m be the number of items and n be the number of bins.

Let *ITEMS* shown in Eq. (2) be a matrix which consists of it_{ij} $(1 \le i \le d, 1 \le j \le m)$. The column number of *ITEMS* denotes the number of each item and the row number denotes the number of dimension corresponding to each item. For example, it_{ij} is the element in row *i* and column *j* which denotes the size of *i-th* dimension in item *j*.

$$ITEMS = \begin{pmatrix} it_{11} & it_{12} & \cdots & it_{1j} & \cdots & it_{1m} \\ it_{21} & it_{22} & \cdots & it_{2j} & \cdots & it_{2m} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ it_{i1} & it_{i2} & \cdots & it_{ij} & \cdots & it_{im} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ it_{d1} & it_{d2} & \cdots & it_{dj} & \cdots & it_{dm} \end{pmatrix} \tag{2}$$

Note that, IT^i is used to represent the column vector *i* i.e. $IT^i = (it_{1i}, it_{2i}, ...it_{di})^T$.

Let *BINS* shown in Eq. (3) be a matrix which consists of b_{ij} $(1 \le i \le d, 1 \le j \le n)$. The column number of *BINS* denotes the number of each bin and the row number denotes the number of dimension corresponding to each bin. For example, b_{ij} is the element in row *i* and column *j* which denotes the size of *i-th* dimension in bin *j*.

$$BINS = \begin{pmatrix} b_{11} & b_{12} & \cdots & b_{1j} & \cdots & b_{1n} \\ b_{21} & b_{22} & \cdots & b_{2j} & \cdots & b_{2n} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ b_{i1} & b_{i2} & \cdots & b_{ij} & \cdots & b_{in} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ b_{d1} & b_{d2} & \cdots & b_{dj} & \cdots & b_{dn} \end{pmatrix} \tag{3}$$

Note that, B^j is used to represent the column vector *j* i.e. $B^j = (b_{1j}, b_{2j}, ...b_{dj})^T$

Definition 3

In this paper, MBP (d>3) only compares the size of every dimension in the items and bins. If the sum of every dimension of the grouping items in each bin is less than or equal to the capacity of bin, IBFD packs the item into the bin, In other words, every element in IT^i must be less than or equal to the corresponding element in B^j. For example, if $IT^i = (0.2, 0.5, 0.3)^T$ and $B^j = (0.5, 0.6, 0.8)^T$, it is obvious that every element in IT^i is less than the corresponding element in B^j, then IBFD packs the item *i* into the bin *j*.

Let *STATES* shown in Eq. (4) be a matrix which consists of s_{ij} $(1 \le i \le m, 1 \le j \le n)$. The column number of *STATES* denotes the number of each bin and the row number denotes the number of each bin. For example, s_{ij} is the element in row *i* and column *j*. If IBFD packs the item *i* into the bin *j*, then $s_{ij} = 1$.

$$STATES = \begin{pmatrix} s_{11} & s_{12} & \cdots & s_{1j} & \cdots & s_{1n} \\ s_{21} & s_{22} & \cdots & s_{2j} & \cdots & s_{2n} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ s_{i1} & s_{i2} & \cdots & s_{ij} & \cdots & s_{in} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ s_{m1} & s_{m2} & \cdots & s_{mj} & \cdots & s_{mn} \end{pmatrix} \tag{4}$$

Definition 4

In IBFD, the sort factor is defined by

$$\begin{aligned} sort\ factor = & \ the\ sum\ of\ every\ dimmension \times \\ & the\ product\ of\ every\ dimmension \end{aligned} \tag{5}$$

For example, if $IT^i = (0.2, 0.5, 0.3)^T$, then

$$\begin{aligned} sort\ factor &= (0.2 + 0.5 + 0.3) \times (0.2 \times 0.5 \times 0.3) \\ &= 0.03 \end{aligned}$$

IBFD packs the items according to the BF after sorting the list of items in non-increasing order and the list of bins in non-decreasing order of each sort factor.

III. DESIGN OF MBPGA FOR MBP

In this section, MBPGA is applied to MBP and makes a comparison with the traditional IBFD.

A. Packing Items Based on IBFD

Let's assume that *ITEMS* shown in Eq. (2) and *BINS* shown in Eq. (3) are already sorted by sort rules which are introduced above, then IT^1 is the largest items and B^1 is the smallest bin in each list. First, IBFD packs IT^1 into first bin that having enough vacant space to pack it, such as B^i. After packing, $B^i = (b_{1i} - it_{11}, b_{2i} - it_{21}, ..., b_{di} - it_{d1})^T$, then IBFD sorts the bins again and packs the next item. Sometimes, the next item (IT^2) cannot packs into B^i for at least one dimension in IT^2

is bigger than the corresponding dimension in B^i. However, considering the item is rotatable, we can rotate IT^2, then IT^2 is packed into B^i. Unfortunately，in IBFD, there doesn't exist a rotate process, so some bins are not filled to the maximum degree. To solve this problem, MBPGA is developed in the next part.

B. Solving MBP by MBPGA

MBPGA is a heuristic algorithm that improves IBFD based on GA.

Definition 5

Let Ω_m denote all the orders of items' with m dimensions for packing. For example, $\Omega_4 = (1,2,3,4)$ is an initial sequence and $\Omega_4 = (2,3,1,4)$ denotes the sequence corresponding to the bin after the process of rotation.

Let IT_i denotes the *i-th* member in Ω_m, such as $\Omega_4 = (2,3,1,4)$, then $IT_2 = 3$.

1) Procedures of MBPGA

To apply the MBPGA to this case, the procedures are as follows [10].

a) Initialize a Population.

Each order of item's dimensions is a member of the population. In MBPGA, the order of item's dimensions is called a chromosome which is coded in real number in this paper. To make it clear for understanding, taking *d*=4 to demonstrate a chromosome. *{3,1,4,2}* is a chromosome which represents the sequence corresponding to the bin.

In this case, because the initial population are generated randomly, there may be some invalid chromosomes such as one dimension is selected for more than one time, for example, *{2,1,2,1}*, *{1,3,3,3}*, and *{4,4,4,4}*.

b) Define the Fitness Function.

How to define fitness function to exclude these invalid chromosomes and, at the same time, to express the fitness value of every member exactly, is a crucial process. This process will be discussed in details later.

c) Calculate the Fitness Value.

d) Selection.

Selecting a new population based on the fitness value above.

e) Crossover.

Single point crossover is applied to this process, such as, two chromosomes *{1,2,5,3}* and *{4,2,3,1}*, after crossover, these two chromosomes may change to *{1,2,3,1}* and *{4,2,5,3}*.

f) Mutation.

In this process, a random probability is occurred to determine whether there exist changes in the chromosome.

g) Repeat the Processes c to f until the Best Packing Solution Is Worked out.

2) Fitness Function for MBPGA

The fitness function of MBPGA takes the role of the natural environment to screening the inferior individuals and retaining the fitted one, so it is a key part in MBPGA.

In this case, fitness function is defined as

$$fitness\_value = \prod_{\substack{i\in[1,m]\\ j\in[1,n]}} sign\left(|B^j - IT^i| + (B^j - IT^i)\right) \cdot sign\left(\prod_{i,j\in\Omega_m} |IT_i - IT_j|\right) \cdot \left(1 / \sum_{\substack{i\in[1,m]\\ j\in[1,n]}} B^j - IT^i\right) \tag{6}$$

In Eq. (6), $\prod_{\substack{i\in[1,m]\\ j\in[1,n]}} sign\left(|B^j - IT^i| + (B^j - IT^i)\right)$ is designed to screening the situation that, at least in one dimension, the item is bigger than the bin. In that situation, there exists $B^j \le IT^i$, then $sign\left(|B^j - IT^i| + (B^j - IT^i)\right) = 0$ as sign(x) is expressed in Eq. (1). The second component is constructed to get rid of the invalid chromosomes. If there exists one or more same dimension numbers in Ω_m, then $sign\left(\prod_{i,j\in\Omega_m} |IT_i - IT_j|\right) = 0$. Finally, $\sum_{\substack{i\in[1,m]\\ j\in[1,n]}} B^j - IT^i$ is the size of the bin after packing, so $\left(1 / \sum_{\substack{i\in[1,m]\\ j\in[1,n]}} B^j - IT^i\right)$ denotes the degree of utilization ratio of the bin. If one of the first two components equal zero, then the fitness value will be zero, in other words, this chromosome will not be remained in the next generation for its fitness value is small.

IV. A PRACTICAL APPLICATION

In this section, a practical application of MBP (d=4, m=2, n=2) is taken as an example to demonstrate the way MBPGA works. Moreover, a comparison is made between the performances of these two methods by applying them to MBP (d=4, m=6, n=4).

ITEMS and *BINS* are initialized as (7) and (8).

$$ITEMS = \begin{pmatrix} 0.15 & 0.2 \\ 0.35 & 0.3 \\ 0.6 & 0.7 \\ 0.9 & 0.8 \end{pmatrix} \tag{7}$$

$$BINS = \begin{pmatrix} 1.8 & 1.2 \\ 1.1 & 1.1 \\ 1.0 & 1.05 \\ 1.2 & 1.8 \end{pmatrix} \tag{8}$$

A. Using IBFD

In this part, IBFD is applied to MBP as follows:

1) Calculate the sort factors of the ITEMS and BINS

$$factor\ of\ IT^1 = (0.15+0.35+0.6+0.9)*(0.15*0.35*0.6*0.9) = 0.0567$$
$$factor\ of\ IT^2 = (0.2+0.3+0.7+0.8)*(0.2*0.3*0.7*0.8) = 0.0672$$
$$factor\ of\ B^1 = (1+1.1+1.2+1.8)*(1*1.1*1.2*1.8) = 12.1176$$
$$factor\ of\ B^2 = (1.05+1.1+1.2+1.8)*(1.05*1.1*1.2*1.8) = 12.848$$

2) Sorting the List of Items in Non-Increasing Order and the List of Bins in Non-Decreasing Order

After sorting, the list of items and bins are shown in (9) and (10).

$$ITEMS = \begin{pmatrix} 0.2 & 0.15 \\ 0.3 & 0.35 \\ 0.7 & 0.6 \\ 0.8 & 0.9 \end{pmatrix} \tag{9}$$

$$BINS = \begin{pmatrix} 1.8 & 1.2 \\ 1.1 & 1.1 \\ 1.0 & 1.05 \\ 1.2 & 1.8 \end{pmatrix} \tag{10}$$

3) Packing

Firstly, IBFD packs IT^2 into B^1, then if B^1 is still having enough vacant space to pack IT^1, IT^1 will be packed into B^1; otherwise IBFD packs IT^1 into B^2.

Now, we pack IT^2 into B^1. After packing, *BINS* changes to (11).

$$BINS = \begin{pmatrix} 1.6 & 1.2 \\ 0.8 & 1.1 \\ 0.3 & 1.05 \\ 0.4 & 1.8 \end{pmatrix} \tag{11}$$

It is obvious that B^1 doesn't have enough vacant space to pack IT^1 for the third and the forth dimensions of B^1 are smaller than IT^1, in this situation, IT^1 is packed into B^2. However, this solution wastes a lot capacity of both two bins.

After packing, IT^2 is packed into B^1 and IT^1 is packed into B^2. Two bins are all used in this packing.

B. Using MBPGA

To minimize the number of used bins, MBPGA is introduced to this case. In MBPGA, the first two steps are the same as the one in IBFD. So, here we start from step3.

Before packing, MBPGA rotates IT^2. After this process, *ITEMS* is shown in (12).

$$ITEMS = \begin{pmatrix} 0.8 & 0.15 \\ 0.3 & 0.35 \\ 0.2 & 0.6 \\ 0.7 & 0.9 \end{pmatrix} \tag{12}$$

Then IT^2 is packed into B^1. After packing, *BINS* changes to (13).

$$BINS = \begin{pmatrix} 1 & 1.2 \\ 0.8 & 1.1 \\ 0.8 & 1.05 \\ 0.5 & 1.8 \end{pmatrix} \tag{13}$$

MBPGA will rotate IT^1 before it is packed. After rotation, *ITEMS* is shown in (14).

$$ITEMS = \begin{pmatrix} 0.8 & 0.9 \\ 0.3 & 0.35 \\ 0.2 & 0.6 \\ 0.7 & 0.15 \end{pmatrix} \tag{14}$$

Now, IT^1 is packed into B^1 for B^1 is still having enough vacant space to pack IT^1.

A conclusion can be drawn from this application that MBPGA is an efficient heuristic algorithm for MBP and it is more effectively than IBFD in this case.

C. The Performances of IBFD and MBPGA

In this part, a practical bin-packing application is introduced to illustrate the performances of IBFD and MBPGA. Table 1 shows the data of 4 bins, including the size of every dimension and the size order of the initial bins.

TABLE 1 DATA OF BINS

Title	Size	Size order (The Min. is 1)
Bin 1	1.0012×1.1888×1.3164×1.6996	1
Bin 2	1.2874×1.4390×1.5431×1.6253	3
Bin 3	1.5017×1.5761×1.7615×1.7624	4
Bin 4	1.1232×1.5044×1.6455×1.7477	2

The data of 6 items are depicted in Table 2, which also consist of the size of every dimension and the size order of the given items.

TABLE 2 DATA OF ITEMS

Title	Size	Size order (The Min. is 1)
Item 1	0.092148×0.14785×0.19817×0.34726	2
Item 2	0.25678×0.43151×0.67227×0.6944	5
Item 3	0.0097586×0.27939×0.53228×0.94623	1
Item 4	0.024855×0.39268×0.67144×0.90644	3
Item 5	0.056933×0.45032×0.83717×0.9715	4
Item 6	0.58247×0.65004×0.68664×0.71943	6

From Table 1 and Table 2, in order to make best use of the given bins, it is obvious that the candidate method prefers to pack the biggest item (Item 6) into the smallest bin (Bin 1) first, if Bin 1 has enough vacant space to pack Item 6. Then, the method follows its packing sequence until the smallest item (Item 3) is packed.

Table 3 shows the experiment results of IBFD and MBPGA, the performances of these two candidate methods are apparent from the results.

TABLE 3 RESULTS OF TWO METHODS

Algorithm	Bin No.	Item No. in each bin	Usage rate of each bin (%)
IBFD	1	3,4	99.59
	2	5	74.72
	3	2	52.82
	4	1,6	91.13
MBPGA	1	1,6	87.85
	2	3,4	89.28
	3	unused	unused
	4	2,5	95.82

IBFD packs all items into the given bins, however, it uses up all available bins. Some of the bins are used inefficiently. Fortunately, MBPGA is applied to this application by designing suitable fitness functions. From Table 3, it is obvious that MBPGA uses only 3 bins to pack all the target items. Moreover, the biggest bin (Bin 3) is never used which can packs other newly arrived items.

V. CONCLUSIONS AND FUTURE WORK

In this paper, a novel concept of sort factor and an improved BFD (IBFD) are proposed. Then, this paper concentrates on designing a new heuristic algorithm to solve the MBP (d>3) by improving the process of IBFD. Besides that, MBPGA is developed to improve the process of IBFD. Moreover, a comparison is made between IBFD and MBPGA to analyze the performances of the two algorithms. Finally, MBPGA is applied to MBP successfully and efficiently.

ACKNOWLEDGMENT

This work is supported by the National Natural Science Foundation of China (NSFC) (71071076). The author gratefully acknowledges the support of the Graduate Innovative Experimental Foundation of NUAA.

REFERENCES

[1] Baker, B.S., “A New Proof for the First-Fit Decreasing Bin Packing Algorithm,” Journal of Algorithms, 6,49-70 (1985).

[2] Johnson, D.S., Demers, A., Ullman, J.D., Garey, M.R., and Graham, R.L., “Worst-case Performance Bounds for Simple One-Dimensional Packing Algorithms,” SZAM *Journal* on *Computing*, 3(4), 299-325 (1974).

[3] Felix T.S. Chan, K.C. Au, L.Y. Chan, T.L. Lau, “Using Genetic Algorithms to Solve Quality-related Bin Packing Problem,” Robotics and Computer-Integrated Manufacturing 23 (2007) 71–81.

[4] A. K. Bhatia, M. Hazra, S. K. Basu, “Better-Fit Heuristic Method for One-dimension BP Problem,” IEEE International Advance Computing Conference, Patiala, India, pp. 193-196, 2009.

[5] E. Hopper, B. C. H. Turton, “A review of the application of meta-heuristic algorithms to 2D regular and irregular strip packing problems,” Artificial Intelligence Review, vol. 16, 257-300, 2001.

[6] Emanuel Falkenauer, “A hybrid grouping genetic algorithm for bin packing,” Journal of Heuristics, vol. 23, pp. 5-30, (1996).

[7] Holland JH. Adaptation in natural and artificial systems. AnnArbor, MI: University of Michigan Press;1975.

[8] E.Falkenauer, “A New Representation and Operators for Genetic Algorithms Applied to Grouping Problems,” Evolutionary Computation, 2(2), 123-144, 1994.

[9] David Simchi-Levi, “New Worst-case Results for the Bin-Packing Problem,” Naval Research Logistics, Vol. 41, pp. 579-585, 1994

[10] Ping Zhang, Ju jiang, Xueshan Han, Zhuoxun Lin, “M-best Subset Selection from N Alternatives Based on Genetic Algorithm,” IEEE Canadian Conference on Electrical and Computer Engineering, Ontario, Canada, pp. 0621-0624, may 2011.