An Estimation of Distribution Algorithm for the 3D Bin Packing Problem with Various Bin Sizes

Yaxiong Cai[1], Huaping Chen[1], Rui Xu[1,*], Hao Shao[2], and Xueping Li[3]

[1] School of Computer Science and Technology,
University of Science and Technology of China,
230026 Hefei, China

[2] School of WTO Research & Education,
Shanghai University of International Business and Economics,
200336 Shanghai, China

[3] Department of Industrial and Information Engineering,
University of Tennessee,
Knoxville, TN 37996-0700, USA

Abstract. The 3D bin packing problem (3DBPP) is a practical problem modeled from modern industry application such as container ship loading and plane cargo management. Unlike traditional bin packing problem where all bins are of the same size, this paper investigates a more general type of 3DBPP with bins of various sizes. We proposed a modified univariate marginal distribution algorithm (UMDA) for solving the problem. A packing strategy derived from a deepest bottom left packing method was employed. The modified UMDA was experimentally compared with CPLEX and a genetic algorithm (GA) approach. The experimental study showed that the proposed algorithm performed better than GA and CPLEX for large-scale instances.

Keywords: Optimization, Bin packing problem, Estimation of distribution algorithm, Genetic algorithm.

1 Introduction

Bin packing, or container loading is very common in industrial applications, such as cutting of foam rubber in arm-chair production, container ship loading,

[*] This work was supported in part by the National Natural Science Foundation of China (No.71171184/61175065/71201151), the Funds for Creative Research Group of China (No. 70821001), the NSFC major program (No.71090401/71090400), the China Postdoctoral Science Foundation (No.2011M501067/2013T60628), the Fundamental Research Funds for the Central Universities (No.WK0110000032), the Program for New Century Excellent Talents in University (No.NCET-12-0512), the Science and Technological Fund of Anhui Province for Outstanding Youth (No.1108085J16), the Humanity and Social Science Youth foundation of Ministry of Education of China (No.13YJC630126), and the Cai Yuanpei Program (No.27927VE).

H. Yin et al. (Eds.): IDEAL 2013, LNCS 8206, pp. 401–408, 2013.

plane cargo management, and warehouse management. With a suitable packing strategy, it is able to reduce cost of transportation or fully utilize the manufacture materials. Another important benefit of minimizing the wasted space in a container is that densely packing container tends to protect the products better and decreases the chance of breakage during shipment. A lot of algorithms and packing strategy had been proposed for the traditional three-dimensional bin packing problem (3DBPP) where all bins are of the same size. However, in practical application we often face the situations that there are bins of different sizes to be used.

This paper investigates a unique variant of 3DBPP. In the investigated problem, there are more than one bin to be used and the bin sizes can differ from each other. Six item orientations are considered. The 3DBPP with various bin sizes contains the classical bin packing problems, where all the bins are of the same size. Unlike the traditional 3DBPP where only the number of bins used needs to be considered, the different sizes of bins pose a greater challenge in providing quality solution. The flexibility of item orientations also expands the search space significantly and hence increases the difficulty of finding optimal solution. This paper proposed an estimation of distribution algorithm for the investigated problem and experimentally tested it. The results showed that the proposed algorithm is a good optimizer for 3dBPP with various bin sizes.

The rest of the paper is structured as follows: The relevant literature is given in Section 2. Section 3 defines the problem precisely and presents the problem formulation. Section 4 presents an estimation of distribution algorithm for the problem. Section 5 conducts the experimental study by comparing the proposed algorithm with CPLEX and a genetic algorithm. Section 6 concludes the paper.

2 Literature Review

The solution techniques in literature for bin packing problems can be classified into three groups. The first group contains algorithms that use mathematical approach and try to reach an exact optimal solution. Martello et al.[2] discussed the lower bound and presented an exact algorithm to pack items into a single bin. Hifi et al.[1] introduced a mixed-integer linear programming formulation (MILP1) for bin packing problem. Heuristics and approximate algorithms belong to the second type. Lim et al.[3] did a comprehensive review on the heuristics that is used for bin packing problem. Most of them are wall or layer-building of the traditional 3DBPP to pack a selected subset of items into a single bin pursuing least waste space. Meta-heuristics such as genetic algorithm belong to the third type. Lodi et al.[4] presented a general Tabu Search technique for the solution of two- and three-dimensional bin packing problems. Crainic et al.[5] presented a new tabu search-based two-level which separates the search for the optimal number of bins from the optimization of the accommodation of items within bins, resulting into a more flexible procedure than the existing ones. Wang et al.[6] presented a hybrid genetic algorithm and a new crossover method for 3DBPP.

3 Problem Description and Formulation

The investigated problem comes from a real world application, where a transportation company tries to make full use of their various bins to pack the cartons which is almost all rectangular. Their goal is to minimize the sum of all the waste space in the bins that are used. Wasted space of a bin is denoted as the volume of the bin minus the sum of the volume of all cartons in the bin.

3.1 Problem Description

There are total m cartons and N boxes, all of which are rectangular. The length, width and height of carton i are denoted as l_i, w_i, h_i, while the length, width and height of bin i is denoted as L_i, W_i, H_i. All the cartons need to be packed into the bins under three constrains.

(1) Each carton lies completely in the bin and does not penetrate the bin's boundary surface. (2) No two cartons in the same bin can overlap with each other. (3) Each carton has only six rotations. Let the length of the bin be X-axis, the width of the bin be the Y-axis and the height of the bin be the Z-axis. Let (l, w, h) means the carton is placed in the way that length of the carton is paralleled to X-axis, width is paralleled to Y-axis and height is paralleled to Z-axis. The six rotation is (l, w, h), (l, h, w), (w, l, h), (w, h, l), (h, w, l), (h, l, w).

3.2 Problem Formulation

In this section, a CPLEX model based on the model introduced by Wu et al.[7] is presented.

Introduced variables:

N: total number of bins

m: total number of cartons

M: a large enough number, an acceptable value is $\sum_{i=1}^{m}(l_i + h_i + w_i) + \sum_{i=1}^{N}(L_i + H_i + W_i)$

(l_i, w_i, h_i): length, width and height of carton i

(L_i, W_i, H_i): length, width and height of bin i

s_{ij} : binary variables, $s_{ij} = 1$ means carton i is in bin j

n_i: binary variables, $n_i = 1$ means bin i is used

x_i, y_i, z_i: continuous variables, coordinate of the front-left-bottom corner point of carton i

$lx_i, ly_i, lz_i, wx_i, wy_i, wz_i, hx_i, hy_i, hz_i$: binary variables, indicating the rotation of carton i. For example, $ly_i = 1$ means that length of carton i is paralleled to X-axis.

$left_{ij}, right_{ij}, front_{ij}, behind_{ij}, below_{ij}, above_{ij}$: binary variables, indicating the relative position between two cartons. For example, $behind_{ij} = 1$ means carton i is behind carton j.

constrains:

for all i in $[1...m]$, k in $[i + 1...m]$

$$x_i + l_i * lx_i + w_i * wx_i + h_i * hx_i \leq x_k + (1 - left_{ik}) * M \tag{1}$$

$$x_k + l_k * lx_k + w_k * wx_k + h_k * hx_k \leq x_i + (1 - right_{ik}) * M \tag{2}$$

$$y_i + l_i * ly_i + w_i * wy_i + h_i * hy_i \leq y_k + (1 - front_{ik}) * M \tag{3}$$

$$y_k + l_k * ly_k + w_k * wy_k + h_k * hy_k \leq y_i + (1 - behind_{ik}) * M \tag{4}$$

$$z_i + l_i * lz_i + w_i * wz_i + h_i * hz_i \leq z_k + (1 - below_{ik}) * M \tag{5}$$

$$z_k + l_k * lz_k + w_k * wz_k + h_k * hz_k \leq z_i + (1 - above_{ik}) * M \tag{6}$$

for all i in $[1...m]$, k in $[i + 1...m]$, j in $[1...N]$

$$left_{ik} + right_{ik} + front_{ik} + behind_{ik} + below_{ik} + above_{ik} \leq s_{ij} + s_{kj} - 1 \tag{7}$$

for all i in $[1...m]$, k in $[i + 1...m]$, j in $[1...N]$

$$\sum_{j=1}^{N} s_{ij} = 1 \tag{8}$$

$$lx_i + ly_i + lz_i = 1 \tag{9}$$

$$wx_i + wy_i + wz_i = 1 \tag{10}$$

$$hx_i + hy_i + hz_i = 1 \tag{11}$$

$$lx_i + wx_i + hx_i = 1 \tag{12}$$

$$ly_i + wy_i + hy_i = 1 \tag{13}$$

for all i in $[1...m]$, j in $[1...N]$

$$x_i + l_i * lx_i + w_i * wx_i + h_i * hx_i \leq L_j + (1 - s_{ij}) * M \tag{14}$$

$$y_i + l_i * ly_i + w_i * wy_i + h_i * hy_i \leq W_j + (1 - s_{ij}) * M \tag{15}$$

$$z_i + l_i * lz_i + w_i * wz_i + h_i * hz_i \leq H_j + (1 - s_{ij}) * M \tag{16}$$

for all j in $[1...N]$

$$\sum_{j=1}^{m} s_i j \leq M * n_j \tag{17}$$

Objective:

$$min\{\sum_{j=1}^{N} L_j * W_j * H_j * n_j - \sum_{i=1}^{m} l_j * w_j * h_j\} \tag{18}$$

Constraints (1) to (6) ensure that any two cartons do not overlap with each other. Constraint (7) ensures that we will only check no overlap constrain when two cartons are placed in the same bin. Constraint (8) ensures that each carton will be placed in exactly one bin. Constraints (9) to (13) ensure that each carton's length, width, and height are paralleled to exactly one axis. Constraints (14) to (16) ensure that none of the cartons penetrate the bin's boundary surface. Constrains (17) ensures that a bin is considered used if any cartons is placed in it.

4 The Proposed EDA

In this section, a modified univariate marginal distribution algorithm (UMDA) for the investigated problem is introduced. This section is structure as: The first subsection introduces the representation strategy for our algorithm. The second subsection describes the detailed steps of the modified UMDA.

4.1 Representation

Representation is a strategy that translates a solution into sequences of numbers or find a suitable solution from number sequences. A good representation can help EDA achieve high quality solution easily. A population has P individuals. We use three number sequences to represent an individual. They are $\{bin_1, ..., bin_N\}$ which is the order of bins to be selected, $\{rotation_1,...,rotation_m\}$ which is the rotation ways of each carton, $\{carton_1,...,carton_m\}$ which is the order of cartons to be packed. The array $\{bin_1, ..., bin_N\}$ is a permutation of $\{1, 2, ..., N\}$. The array $\{carton_1... ,carton_m\}$ is a permutation of $\{1, 2, ..., m\}$. $rotation_i$ is a integer that can be 0, 1, 2, 3, 4 or 5, which means the correspond carton is placed in the dimension of (l, w, h), (l, h, w), (w, l, h), (w, h, l), (h, w, l), (h, l, w) respectively.

Our packing strategy which finds a solution from number sequences is as follows: The cartons are placed into the bins one by one according to the sequence cartons. Carton i uses a rotation of $rotation_i$. The cartons are first tried to be placed into the bin bin_1, and if there is no space for any remaining cartons in bin bin_1, then the bin bin_2 is used. This process will continue, until either all the cartons are placed into the bins or the bins are used up and some cartons are left. In the first case, i.e., all the cartons are placed into the bins, the fitness is calculated as the total volume of the used bins minus the total volume of the cartons; in the second case, i.e., the bins are used up and some cartons are left, the fitness is set to be as the total volume of the used bins. The packing strategy for the single bin we use is similar as the "deepest bottom left first" method[6].

4.2 Generating New Population

The UMDA uses a probability model generated from the existing population to sample new populations. If the same method is applied for 3DBPP, we will need to learn the distribution of the three arrays and sample from the distributions separately. However, since the array $\{rotation_i\}$ are rotation ways of the array $\{carton_i\}$, sampling them separately will break the connection between them and lead to poor performance. We modified UMDA so that we sample new individuals by randomly picking genes from existing individuals which will not break the connection between array $\{rotation_i\}$ and $\{carton_i\}$. The detailed steps are as follows.

To sample the kth ordered variable in array $\{rotation_i\}$ and $\{carton_i\}$, a random number rnd between 0 and P is generated. If the variable in the kth position of array $\{carton_i\}$ in the rndth ordered individual has not appeared in the sampled individual before, the sample individual's kth ordered variable in array $\{carton_i\}$ and $\{rotation_i\}$ equals to the rndth individual in existing population. If it had appeared before, we check the population in order, starting from individual *rnd*, until one individual with the kth ordered gene never appearing before is met or all individuals are checked. If all individuals are checked, a suitable random value in generated for position k for array $\{rotation_i\}$ and $\{carton_i\}$. The pseudo code for sampling array $\{rotation_i\}$ and $\{carton_i\}$ are as follows. The same steps are followed to sample array $\{bin\}$. After the three number sequences are sampled, we conduct mutation to them and then we finish sampling an individual.

The process of sampling array carton and array rotation

```
Sampling-individual( Pop[1..P]
    const
        P,N: Int;
    var
        i,j,temp,rnd: Int;
    begin
        i :=0;
        j :=0;
        repeat
            rnd := random(0,P);
            If pop[rnd].carton[j] never appeared
                individual.carton[j] := pop[rnd].carton[j];
                individual.rotation[j] := pop[rnd].rotation[j];
                return;
            Else temp :=rnd;
                repeat
                    temp :=temp+1;
                    if pop[temp].carton[j] never appeared
                        individual.carton[j] := population[temp].carton[j];
                        individual.rotation[j] := population[temp].rotation[j];
                        return
                until temp mod N ==rnd mod N
                assign carton[j] and rotation[j] randomly
            until j>=N
end.
```

For each generation, E best individuals are selected and added to the next population directly. $P-E$ individuals are sampled using the strategy mentioned above. After enough generations, a good enough solution can be achieved.

In the worst case, the time complexity of generating a gene is $O(P)$, thus the time complexity of of generating an individual is $O(m*P+N*P)$. In our algorithm, we generate P individuals for *gen* generations, so the time complexity of the algorithm is $O(gen*P*(m*P+N*P))$.

5 Numerical Examples and Analysis

In the following experimental study, we compared our algorithm with a GA approach and the CPLEX solver. The GA approach uses the same representation strategy with our EDA and employs the crossover method mentioned in [6]. After the crossover is done, we conduct mutation operation to the number sequences. For every number in array $\{bin\}$ and $\{rotation\}$, it is switched position with another randomly selected number with probability pm. For every number in array $\{rotation\}$, it is changed to its another possible value with probability pm. Each population contains P individuals. Before generating the next population, E best individuals are directly added to the next population. In the experiment, the parameters for GA and modified UMDA are set as: $pm = 0.01$, $P = 100$, $E = 10$, $gen = 100$. The meaning of pm, P, E is in the above algorithm description. gen stands for generation which we repeat gen times of generating new population until we stop. The instances we use are from a transportation company. Each instance is tested for 10 times. The data shown in the Table 1 is the space utilization rate, which is calculated by $space_{items}/(space_{items} + space_{wasted})$. A larger value of the rate indicates a better quality of the solution. The time limit we set for CPLEX is 1800 seconds. If CPLEX failed to get a solution in 1800 seconds, we use a "-" to denote it.

Table 1. Comparison of space utilization percentage among three approaches

Instances	GA average	GA best	EDA average	EDA best	CPLEX	CPLEX gap
12-1	0.73	0.73	0.53	0.71	0.73	0
12-2	0.82	0.85	0.59	0.67	0.85	0
12-3	0.49	0.49	0.47	0.49	0.49	0
12-4	0.27	0.45	0.19	0.45	0.45	0
12-5	0.66	0.66	0.49	0.66	0.66	0
20-1	0.77	0.82	0.7	0.82	0.58	99%
20-2	0.76	0.77	0.67	0.77	0.62	99%
20-3	0.77	0.8	0.66	0.78	0.63	99%
20-4	0.68	0.72	0.6	0.77	0.7	98%
20-5	0.71	0.73	0.72	0.88	0.65	99%
50-1	0.69	0.78	0.74	0.83	-	-
50-2	0.7	0.76	0.74	0.83	-	-
50-3	0.68	0.73	0.75	0.85	-	-
50-4	0.72	0.76	0.73	0.8	-	-
50-5	0.62	0.7	0.66	0.75	-	-
78-1	0.65	0.68	0.7	0.74	-	-
78-2	0.66	0.69	0.71	0.74	-	-
78-3	0.65	0.68	0.74	0.82	-	-
78-4	0.67	0.7	0.7	0.74	-	-
78-5	0.67	0.72	0.72	0.79	-	-

From the experimental data, the following observations can be concluded: (1) For small instances with 12 items, CPLEX and GA are very likely to solve to optimal; modified UMDA can achieve optimal solution but its performance is not very stable. (2) For medium instances with 20 items, CPLEX can not find a good solution in 1800s. Both GA and modified UMDA can find a good solution. GA has a better "average solution" and modified UMDA has a better "best solution". (3) For large instances with 50 or 78 items, CPLEX failed to find a solution. Both GA and modified UMDA are able to find a solution. Modified UMDA has a better "average solution" and "best solution".

From the observations, it can be seen that the modified UMDA is a good choice for large instances. For small and medium instances, although the modified UMDA do not perform very stable, we can still get a solution better than or at least as good as GA and CPLEX by repeating the algorithm on the same instance for several times and pick out the best solution.

6 Conclusion

In this paper, bin packing problems with various bin sizes are investigated. The mathematical formulation and an estimation of distribution algorithm for the problem are proposed and experimentally studied. In the proposed algorithm, a packing method for multi bins is used for representation and a modified univariate marginal distribution algorithm is used for generating new population. The experiment results show that our proposed method seems to be a good optimizer for 3DBPP with various bin sizes, especially for a large instance with 50 or more items to pack.

References

1. Hifi, M., Kacem, I., Nègre, S., et al.: A linear programming approach for the three-dimensional bin-packing problem. Electronic Notes in Discrete Mathematics 36, 993–1000 (2010)
2. Martello, S., Pisinger, D., Vigo, D.: The three-dimensional bin packing problem. Operations Research 48(2), 256–267 (2000)
3. Lim, A., Rodrigues, B., Yang, Y.: 3-d container packing heuristics. Applied Intelligence 22(2), 125–134 (2005)
4. Lodi, A., Martello, S., Vigo, D.: Tspack: a unified tabu search code for multi-dimensional bin packing problems. Annals of Operations Research 131(1-4), 203–213 (2004)
5. Crainic, T., Perboli, G., Tadei, R.: TS2PACK: A two-level tabu search for the three-dimensional bin packing problem. European Journal of Operational Research 195(3), 744–760 (2009)
6. Wang, H.F., Chen, Y.J.: A hybrid genetic algorithm for 3d bin packing problems. In: 2010 IEEE Fifth International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA), pp. 703–707. IEEE (2010)
7. Wu, Y., Li, W.K., Goh, M., Souza, R.: Three-dimensional bin packing problem with variable bin height. European Journal of Operational Research 202(2), 347–355 (2010)
8. Henrion, M.: Propagating uncertainty in bayesian networks by probabilistic logic sampling. In: Uncertainty in Artificial Intelligence, vol. 2, pp. 149–163 (1988)
9. Mühlenbein, H.: The equation for response to selection and its use for prediction. Evolutionary Computation 5(3), 303–346 (1997)