A Mathematical Model And Solution Algorithm of the Bin Packing Problem with Group Constraints

F. Nuriyeva, B. Tezel, E. Nasiboğlu

The Bin Packing Problem (BPP) is to find the minimum number of bins needed to pack a given set of objects of known sizes so that they do not exceed the capacity of each bin. In this paper, a new version of the bin packing problem and a solution approach based on the dynamic programming for this problem is proposed. The algorithm is coded in C. The working principle of the algorithm is shown as an example. Computational experiments show the efficiency of the proposed algorithm.

Keywords: bin packing problem, dynamic programming.
ACM 2012: Mathematics of computing → Discrete mathematics → Combinatorics → Combinatorial optimization.
MSC 2010: 90C39, 68R05, 97M40
UDC: 519.7, 519.8

Introduction

The bin packing problem can be defined as follows: a set of n objects each with a given weight (or size) $(w_i > 0, i = 1 \dots n)$ has been given. We want to place these objects into bins of a given capacity $C(C > w_i, i = 1 \dots n)$ so that the total number of bins needed is minimized. This problem has many practical applications: Vehicles such as are pallets, containers, trailers, trucks, rail cars, ships and so on, are to be loaded with different items. The aim is to use as few vehicles as possible to carry the loads without exceeding the capacity of each vehicle. Another example is where tubes or cables are to be cut from quantities of standard length C. We want to use as few tubes or cables of standard length as possible to meet the demand. The same idea is used in metal working where steel sheets of different sizes must be cut from "master" sheets. Yet another example is in scheduling, where tasks of varying duration must be allocated using the least number of machines or processors [1], [2].

The contributions of this paper are threefold. We introduce a new packing problem that is both relevant for many transportation and logistics planning problems, especially freight shipping, when a company has to ship orders to different customers. Some of these orders may be urgent the other may wait before shipped. Also, different orders can be placed into the same shipping box with a certain condition. For example, containers which are loaded to ship, can be closed or opened, designed for food or liquids [3]. We present a formalization for the proposed problem.

The paper is organized as follows. Section 1 presents a mathematical model of the proposed problem, an algorithm for solving the proposed problem is introduced in section 2, an example of the problem is presented in section 3, section 4 concludes the paper.

Problem Definition and Formulation

Assume that we have n set of objects, X, which joined into different groups $l \in L$ according to particular features and characterized by volume (w) and profit (p). Also we have a set of bins (y) with given volume (v) which create constraints based on groups of objects and has its own constraint of upper bound of total volume of bins (V), and cost (c). The bins are separated into types with different upper availability limits (U_t). Part of the object, which denoted compulsory (C), must be loaded, while a selection has to be made among the non-compulsory (NC) objects. The object is to minimize difference between two total cost of the used bins and total profit of loaded objects which are non-compulsory [4], [5].

Let I denote the set of n items, with the volume (w_i) and the profit (p_i) of item $i \in I$. $I^c \subseteq I$ the subset of items define absolutely loaded objects and $I^{NC} = I \backslash I^c$ the subset of items which may be chosen if profitable. Let J denote the set of available containers and let T be the set of container types. For any bin $j \in J$, let $\sigma(j) \in T$ be the type of bin j.

Each of the object belong to a group and the volume of this group in the container is limited.

$$\sum_{i \in I} w_i x_{ijl} \leq v_{lj}, \ j \in J, \ l \in L;$$

The volume of container, which will be used, is restricted.

$$\sum_{j \in J} \sum_{l \in L} y_j v_{jl} \leq V$$

Following constraints ensure that each compulsory and non-compulsory objects is loaded into exactly one and at most one container.

$$\sum_{l \in L} x_{ijl} \leq 1, \ i \in I, \ j \in J;$$

$$\sum_{j \in J} x_{ijl} = 1, \ i \in I^C, \ l \in L;$$

$$\sum_{j \in J} x_{ijl} \leq 1, \ i \in I^{NC}, \ l \in L.$$

The bins are separated into types with different upper availability limits. Maximum number of available container is enforced by,

$$\sum_{j \in J : \sigma(j)} y_j \leq U_t, \ t \in T.$$

Object assignment binary variables and bin selection binary variables is as follows, respectively.

$x_{ijl} = 0$ if not assigned to any container and $x_{ijl} = 1$ if assigned to any container.

$y_j = 0$ if not selected to assignment and $y_j = 1$ if selected to assignment.

Thus, a mathematical model of the proposed problem is as follows:

$$Min\{\sum_{j\in J} c_j y_j - \sum_{j\in J}\sum_{l\in L}\sum_{i\in I^{NC}} p_i x_{ijl}\} \tag{1}$$

Subject to,

$$\sum_{i\in I} w_i x_{ijl} \le v_{lj},\ j \in J,\ l \in L;$$
$$\sum_{j\in J}\sum_{l\in L} y_j v_{jl} \le V;$$
$$\sum_{l\in L} x_{ijl} \le 1,\ i \in I,\ j \in J;$$
$$\sum_{j\in J} x_{ijl} = 1,\ i \in I^C,\ l \in L;$$
$$\sum_{j\in J} x_{ijl} \le 1,\ i \in I^{NC},\ l \in L;$$
$$\sum_{j\in J:\sigma(j)} y_j \le U_t,\ t \in T;$$
$$y_j \in \{0,1\},\ j \in J;$$
$$x_{ijl} \in \{0,1\},\ i \in I,\ j \in J,\ l \in L.$$

The objective function (1) minimizes the total net cost, which is calculated by difference between two total cost of the used bins and total profit of loaded objects which are non-compulsory. The profit of the compulsory objects is not included because it is compensated to a constant.

An Algorithm for Solving the Proposed Problem

To find an admissible solution of the problem, tthe following algorithm based on dynamic programming is proposed. Dynamic programming consists of considering n stages (for m increasing from 1 to n) and computing, at each stage $m > 1$, the values (for d increasing from 0 to c) using the classical recursion [6].

The steps of the algorithm is as follows.

Separate objects into compulsory and non-compulsory;
Separate both compulsory and non-compulsory objects by groups;
while *All compulsory objects are not assigned, the total volume of containers is not exceed constraint of upper bound of total volume of containers and containers that can be used is suitable* **do**
    Calculate sum of $\frac{p_i}{w_i}$ for each group according to compulsory unassigned objects;
    Sort sums by descending order;
    Determine group of objects, which has the maximum sum and select a container which has the maximum volume of this group in the container;
    if *this container can not be suitable* **then**
        Select next container which has the maximum volume of this group in the container;
    end
    Fill the container with objects of each group according to available space by dynamic programming;
end
if *All compulsory objects are not assigned* **then**
    return Solution is not exist;
end
Fill the remaining space of used containers with non-compulsory objects by dynamic programming respectively;
while *Sum of p_i of used objects in assignment is bigger than c_i of used container, all non-compulsory objects are not assigned, the total volume of containers is not exceed constraint of upper bound of total volume of containers and containers that can be used is suitable* **do**
    Calculate sum of $\frac{p_i}{w_i}$ for each group according to non-compulsory unassigned objects;
    Sort sums by descending order;
    Determine group of objects, which has the maximum sum and select a container which has the maximum volume of this group in the container;
    if *this container can not be suitable* **then**
        Select next container which has the maximum volume of this group in the container;
    end
    Fill the container with objects of each group according to available space by dynamic programming;
    return Solution ;
end

Algorithm 1: Algorithm for solving the proposed problem

An Example

The following example demonstrates the execution of one cycle of the proposed algorithm. It is required to place 40 objects in to three type container. 14 of Objects are denoted compulsory, 26 of objects are denoted non-compulsory (Table 2). Amount of the each type of container is two. Volume and cost of the container according to groups of the objects is given in Table 1. When algorithm is applied, all compulsory objects are assigned and assigned non-compulsory objects are shown in Table 3. Finally, the obtained value of objective function is found as 576.

Example which is randomly generated, has been created in C program. The weights w_i, the profits p_i, are uniformly random distributed. Also, Obligation and groups are randomly selected, too.

Table 1. Volume and cost of the container according to groups of the objects

Type of Container	Volume of the container according to groups of the objects			Amount	Cost
	1	2	3		
1	100	50	50	2	200
2	50	100	50	2	200
3	50	50	100	2	200

Table 2. Data of example

Object	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Volume	21	32	23	46	20	10	5	44	25	46	11	40	22	16
Profit	46	24	42	16	31	17	11	47	14	38	27	24	16	46
Obligation	0	0	0	0	1	0	0	1	0	1	1	1	0	0
Group	3	1	1	2	2	2	2	2	1	1	2	3	3	1

Table 2 (Continued)

Object	15	16	17	18	19	20	21	22	23	24	25	26	27	28
Volume	15	3	24	40	23	9	13	24	28	39	25	20	34	41
Profit	11	6	43	1	23	14	4	27	6	26	6	42	50	22
Obligation	1	1	0	0	0	1	0	1	1	0	1	0	0	0
Group	3	1	3	2	3	2	3	3	2	1	1	3	2	2

Table 2 (Continued)

Object	29	30	31	32	33	34	35	36	37	38	39	40
Volume	46	48	28	33	29	33	23	44	17	47	47	43
Profit	27	35	33	21	50	34	32	21	2	24	10	17
Obligation	0	0	1	0	0	0	0	0	0	1	0	1
Group	3	3	1	1	3	1	3	1	2	3	2	2

Table 3. Assigned non-compulsory objects

Object	1	2	3	6	7	9	14	17	19	26	27	33	34	37
Volume	21	32	23	10	5	25	16	24	23	20	34	29	33	17
Profit	46	24	42	17	11	14	46	43	23	42	50	50	34	2
Group	3	1	1	2	2	1	1	3	3	3	2	3	1	2

Conclusion

As a result, a new problem which is inspired by bin packing problem, is proposed. A mathematical model of the proposed problem is developed. Furthermore, a new algorithm which is based on dynamic programming is suggested.The algorithm is coded in C language and tested on an example. The result shows that proposed algorithm is efficient.

Acknowledgement

F. Nuriyeva was partially supported by TUBITAK 2216 program.

References

[1] N. Mohamadi, "Application of genetic algorithm for the bin packing problem with a new representation scheme," *Mathematical Sciences*, vol. 4, no. 4, pp. 253–266, 2010.

[2] M. E. Berberler and U. G. Nuriyev, "A new heuristic algorithm for the one-dimensional cutting stock problem," *An International Journal Applied and Computational Mathematics*, vol. 9, no. 1, pp. 19–30, 2010.

[3] M. M. Baldi, T. G. Crainic, G. Perboli, and R. Tadei, "The generalized bin packing problem," tech. rep., Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation, CIERRELT, 2011.

[4] F. Nuriyeva, B. T. Tezel, and E. Nasiboğlu, "A methematical model of the multicriteria limited bin packing problem with fuzzy qualities," in *Proceedings of the "Caucasian Mathematics Conference CMC I"*, pp. 145–146, 2014.

[5] E. Nasiboğlu, "An algorithm of constructing an admissible solution to the bin packing problem with fuzzy constraints," *International Journal of Computer and Systems Sciences*, vol. 43, no. 2, pp. 205–212, 2004.

[6] S. Martello and P. Toth, *Knapsack Problems Algorithms and Computer Implementations*. John Wiley & Sons, 1990.

Authors

Fidan Nuriyeva — Dr., Dokuz Eylul University, Faculty of Science, Department of Computer Science, Izmir, Turkey, Senior Researcher, Institute of Cybernetics of ANAS, Baku, Azerbijan; E-mail: *nuriyevafidan@gmail.com*

Barış Tezel — Research Assistant, Dokuz Eylul University, Faculty of Science, Department of Computer Science, Izmir, Turkey; E-mail: *baris.tezel@deu.edu.tr*

Efendi Nasiboğlu — Professor, Dokuz Eylul University, Faculty of Science, Department of Computer Science, Izmir, Turkey; E-mail: *efendi.nasibov@deu.edu.tr*