Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in DISCRETE MATHEMATICS

Electronic Notes in Discrete Mathematics 58 (2017) 39–46

www.elsevier.com/locate/endm

VNS matheuristic for a bin packing problem with a color constraint

Y. Kochetov [a,1,3] A. Kondakov [b,2]

[a] *Sobolev Institute of Mathematics, Novosibirsk, Russia*
Novosibirsk State University, Novosibirsk, Russia

[b] *Novosibirsk State University, Novosibirsk, Russia*

Abstract

We study a new variant of the bin packing problem. Given a set of items, each item has a set of colors. Each bin has a color capacity, the total number of colors for a bin is the union of colors for its items and can not exceed the bin capacity. We want to pack all items into the minimal number of bins. For this NP-hard problem we apply the column generation technique based on the VNS matheuristic for the pricing problem. To get optimal or near optimal solutions we apply VNS matheuristic again using optimal solution for the large scale linear programming relaxation. Computational experiments are reported for the randomly generated test instances with large bin capacity and number of items up to 250.

Keywords: Matheuristic, local search, large neighborhood, column generation

[1] Email: jkochet@math.nsc.ru
[2] Email: tyxtyxyc@gmail.com
[3] This research was supported by RSF grant 15-11-10009.

http://dx.doi.org/10.1016/j.endm.2017.03.006

1 Introduction

In the classical bin packing problem, a set of weighted items must be packed in minimal number of identical bins such that the sum of weights of items in a bin does not exceed the bin's capacity. In this paper we study a new variant of the bin packing problem. Each item has not weight but has some colors. The bin capacity limits the total number of colors for its items. The goal is to pack all coloring items in the minimal number of identical bins such the the total number of colors of items in a bin does not exceed the bin capacity. It is NP-hard problem and it can be reformulated as a biclique vertex covering problem for bipartite graphs [5].

The bin packing problem with color constraints (CPP) is a relatively recent line in combinatorial optimization. One of the applications of the problem is beverage package printing [10]. In [12] the bin packing problem with classes of items (colors) is used to model video-on-demand applications. A biclique covering (biclustering), which is a straight reformulation of CPP, is used to model protein-protein interaction [4].

There are some versions of the problem with online and offline settings [2,11]. In the colored bin packing [3], each bin has a maximum color capacity, i.e. a limit on the number of items of a particular color. This version is originated in the production planning of a steel plant. In a more generalized version of the color bin packing problem [8,9] the constraints among items are described by a conflict graph. In the black and white bin packing problem with alternation constraints [1], two items with the same color can not be packed adjacently to each other.

In this paper we consider a new version of the colored bin packing problem which is a special case of the co-printing problem [10]. We assume that each item has zero weight and arbitrary number of colors. We apply the column generation technique to get lower bound for number of bins in optimal solution and VNS matheuristic for the pricing problem. We use this heuristic to get upper bound as well. Solutions for the linear programming relaxation are rounding down and the remaining items are packed by the VNS matheuristic.

The paper is organized as follows. In Section 2 we introduce notations and present the mathematical model. In Section 3 we describe the pricing problem and discuss its complexity. In Section 4 we define three types of neighborhoods and present a pseudocode of the VNS matheuristic for the pricing problem. In Section 5 we design a heuristic for the CPP problem. Finally, in Section 6 we discuss some preliminary experimental results for randomly generated test instances with large bin capacity and number of items up to 250.

2 Mathematical Model

Let us introduce the following notations:
$I = \{1, \ldots, n\}$ is the set of items;
$J = \{1, \ldots, m\}$ is the set of colors;
$K_i \subset J$ is the set of colors for item i;
b is the number of different colors for each bin;
$p = (p_1, \ldots, p_i, \ldots, p_n)$ is a bin pattern (or bin for shot) where $p_i \in \{0, 1\}$ denotes whether item i is in the bin or not;
$P = \{p : \mid \cup_{i \in I}(K_i : p_i = 1) \mid \leq b\}$ is the set of all feasible bins.

Decision variables:
$y_p = 1$ if bin p is used in the solution and $y_p = 0$ otherwise.

Now we can write the CPP as follows:

$$\min \Big\{ \sum_{p \in P} y_p \ : \ \sum_{p \in P} p_i y_p \geq 1, i \in I, \ y_p \in \{0, 1\} \Big\}.$$

In this linear integer programming formulation we have a lot of variables and a few constraints. The large scale formulations allow us to exclude symmetries which usually presented in the bin packing compact representations. Moreover, large scale formulations as a rule have small integrality gap and the linear programming relaxation can be solved exactly by column generation technique. Below we apply this approach and use VNS and matheuristic ideas for accelerating method.

3 Pricing problem

According to the classical column generation approach we select a subset of bins $P' \subset P$, remove the integrality constraints and solve the dual linear programming problem:

$$\max \Big\{ \sum_{i \in I} w_i \ : \ \sum_{i \in I} p_i w_i \leq 1, p \in P', \quad w_i \geq 0 \Big\},$$

where the dual variable w_i can be considered as a price for item i. To enlarge the subset P' or terminate the method, we should solve the following pricing problem with optimal values $\overline{w}_i$ of the dual variables.

Let us introduce additional variables:
$x_i = 1$ if item i is placed in a bin and $x_i = 0$ otherwise.
$z_j = 1$ if a bin contains item with color j and $z_j = 0$ otherwise.

Then, the pricing problem can be stated as follows:

$$\min(1 - \sum_{i \in I} \overline{w}_i x_i)$$

$$\text{s.t.} \sum_{j \in J} z_j \leq b;$$
$$x_i \leq z_j, \quad j \in K_i, i \in I;$$
$$x_i, z_j \in \{0, 1\}, \quad i \in I, j \in J.$$

The objective function minimizes the reduced cost. The first constraint controls the total number of colors for a bin. The second constraint shows the relations between items and colors. It is easy to see that a knapsack problem can be reduced to the pricing problem, and as a result, the pricing problem is NP-hard. We have to solve a lot of the pricing problems with different prices. Thus, we design a VNS matheuristic to find optimal or near optimal solutions.

4 VNS matheuristic

Variable Neighborhood Search is the well-known metaheuristic for combinatorial and global optimization based upon systematic change of neighborhood within local search [7]. Below we apply this approach to the pricing problem. For a feasible solution (x_i^0, z_j^0), we put $K^0 = \cup\{K_i \mid x_i^0 = 1\}$ and define three types of neighborhoods.

- k-ItemAdd neighborhood. We collect all items with at most k additional colors to K^0 and select one of them, say i'. Then we create a subset $K' = K^0 \cup K_{i'}$ and a subset of items $I' = \{i \in I \mid K_i \subseteq K'\}$. The neighboring solution is defined as the optimal solution to the pricing problem with the following restriction: $x_i = 0, i \notin I'$. The size of the neighborhood is $O(n)$.
- k-ColorDel neighborhood. We select a subset of colors $K' \subseteq K^0, |K'| \leq k$. The neighboring solution is defined as the optimal solution to the pricing problem with the following restriction: $z_j = 1, j \in K^0 \setminus K'$. The size of the neighborhood is exponential.
- k-ColorSwap neighborhood. We remove k colors from the set K^0 and include other k colors resulting in a set K'. The neighboring solution is defined as all items with colors in K'. The size of the neighborhood is exponential, but each neighboring solution can be evaluated in polynomial time.

To get neighboring solutions for the k-ItemAdd and k-ColorDel neighborhoods, we need to solve the pricing problems. For small k they have small

dimension and can be solved easily by Gurobi software [6]. Resolution of pricing problem in full dimension is time consuming.

For diversification, we collect the subsets K' during the search by the k-ItemAdd neighborhood and include the cuts

$$\sum_{j \in K'} z_j \leq b - 1$$

into the pricing problem for removing the previously visited solutions. For the k-ColorDel neighborhood we use the cuts

$$\sum_{j \in J \setminus (K^0 \setminus K')} z_j \leq k - 1.$$

The cuts are accumulated in the TabuList with fixed length. The pseudocode of the matheuristic is presented below.

VNS matheuristic

1. Select the set of neighborhood structures for $k = 1, \ldots, k_{max}$; find an initial solution (x, z); put TabuList := $\emptyset$; choose a stopping condition;

2. Repeat the following sequence until the stopping condition is met:

(1) Set $k := 1; t := 1$;

(2) Repeat the following steps until $k = k_{max}$:

(a) Select a solution (x', z') at random by the k-ItemAdd neighborhood for even t and by the k-ColorDel neighborhood for odd t;
(b) Update TabuList; $t := t + 1$;
(c) If solution (x', z') is better than incumbent solution (x, z) then move $(x, z) := (x', z')$ and $k := 1$; else $k := k + 1$;
(d) If we have no improvement for $\frac{k_{max}}{2}$ iterations then move by the k-ColorSwap neighborhood at random direction;

3. Return the best found solution (x, z).

The items with large color sets K_i are the most inconvenient for the heuristic. To improve its efficiency, we select all large items, $|K_i| \geq b - r$, $1 \leq r \leq 10$, and solve the pricing problem for each of them with additional constraint

$x_i = 1$. Then we apply the VNS matheuristic for the remaining items. The best found solution is returned as a result of the modified algorithm.

We use this approach at each iteration of the column generation procedure and include all solutions with negative reduced cost into the restricted master problem. In such a way we decrease the number of iterations but increase the total number of columns. If the VNS matheuristic cannot find a solution with negative reduced cost, then we solve the pricing problem to optimality by commercial solver Gurobi.

5 Heuristic for the CPP

We apply three heuristics to find optimal or near optimal solutions:

- The FFD heuristic. It is adaptation of the well known FFD algorithm for the classical bin packing problem. The items are sorted by the nonincreasing of the cardinality of their color sets. Then we put each item in the first bin in which it fits. If it does not fit in any bin, we open a new bin.
- The FillBin heuristic. We put $\overline{w}_i = |K_i|^2$ and use these artificial prices in the pricing problem. The VNS matheuristic is applied to fill the first bin. The packed items are removed and VNS matheuristic is applied again until all items are packed into bins.
- The LP heuristic. We use the following idea [10] at the last iterations of the column generation procedure. We take the round down LP solution to the restricted master problem as a partial solution and apply the FillBin heuristic to the remaining items. The best found solution is returned as the result of the LP heuristic.

Note that the last LP heuristic is time consuming but produces the most strong solutions. Moreover, we can use it for large scale instances as well if the column generation procedure is terminated in an intermediate iteration. Sure, in such a case we have not a lower bound for the global optimum. The FFD and FillBin heuristics are fast but produce weak solutions. We use these heuristics at the initialization step of the VNS and column generation procedure.

6 Computational experiments

We conduct our computational experiments for the randomly generated test instances in 8 sets of parameters n, m and b. For each set we generate 10 instances. Thus, the total amount of tests is equal to 80. The color set K_i for

each item is generated by the following procedure. We uniformly choose an integer l from 1 to b and uniformly choose a 0-1 vector with exactly l ones.

Table 1 presents the results of our experiments. First three columns show the parameters of the instances. Column $Iter$ shows the number iterations for the column generation procedure. Column $Cols$ shows an amount of generated columns. Columns VNS and Gur show the total running time (in seconds) for the VNS matheuristic and Gurobi solver respectively. Column $Time$ shows the total running time for the method. Columns LB and UB present the lower bound of the linear programming relaxation and the best upper bound for the global optimum.

We note that the performance of the method strongly depends on the integrality gap and the bin capacity. For instances without the gap, our approach shows good performance both in running time and the quality of the solutions. It seems that the uniformly generated instances are quite *easy* for the method. Many of them have not the integrality gap in the large scale reformulation. If the integrality gap is positive, we need a lot of efforts to solve the linear programming relaxation. Nevertheless, we can apply the LP heuristic in such a case as well even for large bin capacity. Note that previous studies [10] deal with small bin capacity only, $b \leq 10$.

n	m	b	$Iter$	$Cols$	VNS	Gur	$Time$	LB	UB
150	55	30	78	351	1303	755	2058	53	53
150	55	30	128	458	1794	3622	5416	52	54
200	90	40	31	132	191	182	373	82	82
200	90	40	87	242	1692	1785	3477	86	86
225	90	40	70	301	751	628	1379	91	91
225	90	40	57	221	676	1697	2373	95	96
250	90	40	88	394	1805	2980	4785	102	102
250	90	40	175	610	4376	13227	17603	109	112

Table 1
Computational results

References

[1] Balogh, J., J. Békési, G. Dosa, H. Kellerer and Z. Tuza, *Black and white bin packing*, in: *Approximation and Online Algorithms*, Springer, 2012 pp. 131–144.

[2] Böhm, M., J. Sgall and P. Veselỳ, *Online colored bin packing*, in: *Approximation and Online Algorithms*, Springer, 2014 pp. 35–46.

[3] Dawande, M., J. Kalagnanam and J. Sethuraman, *Variable sized bin packing with color constraints*, Electronic Notes in Discrete Mathematics **7** (2001), pp. 154–157.

[4] Ding, C., Y. Zhang, T. Li and S. R. Holbrook, *Biclustering protein complex interactions with a biclique finding algorithm*, in: *6th International Conference on Data Mining (ICDM'06)*, IEEE, 2006, pp. 178–187.

[5] Fleischner, H., E. Mujuni, D. Paulusma and S. Szeider, *Covering graphs with few complete bipartite subgraphs*, Theoretical Computer Science **410** (2009), pp. 2045–2053.

[6] Gurobi Optimization, I., *Gurobi optimizer reference manual* (2015).
URL http://www.gurobi.com

[7] Hansen, P. and N. Mladenović, *Variable neighborhood search: Principles and applications*, European Journal of Operational Research **130** (2001), pp. 449–467.

[8] Jansen, K., *An approximation scheme for bin packing with conflicts*, Journal of Combinatorial Optimization **3** (1999), pp. 363–377.

[9] Muritiba, A. E. F., M. Iori, E. Malaguti and P. Toth, *Algorithms for the bin packing problem with conflicts*, Informs Journal on Computing **22** (2010), pp. 401–415.

[10] Peeters, M. and Z. Degraeve, *The co-printing problem: A packing problem with a color constraint*, Operations Research **52** (2004), pp. 623–638.

[11] Shachnai, H. and T. Tamir, *Polynomial time approximation schemes for class-constrained packing problems*, Journal of Scheduling **4** (2001), pp. 313–338.

[12] Xavier, E. C. and F. K. Miyazawa, *The class constrained bin packing problem with applications to video-on-demand*, Theoretical Computer Science **393** (2008), pp. 240–259.