Available online at www.sciencedirect.com
ScienceDirect
Electronic Notes in DISCRETE MATHEMATICS
Electronic Notes in Discrete Mathematics 55 (2016) 61–64
www.elsevier.com/locate/endm

Column generation for the variable cost and size bin packing problem with fragmentation

Marco Casazza [1]

UPMC University Paris 06, UMR 7606, LIP6, F-75005, Paris, France

Alberto Ceselli [2]

Dipartimento di Informatica, Università Degli Studi di Milano, Crema, Italy

Abstract

Bin Packing Problems with Item Fragmentation (BPPIF) are variants of classical Bin Packing in which items can be split at a price. We extend BPPIF models from the literature by allowing a set of heterogeneous bins, each potentially having a different cost and capacity. We introduce extended formulations and column generation algorithms to obtain good bounds with reasonable computing effort. We test our algorithms on instances from the literature. Our experiments prove our approach to be more effective than state-of-the-art general purpose solvers.

Keywords: Bin Packing, Item Fragmentation, Variable Cost and Size, Column Generation.

1 Introduction

Bin Packing Problems with Item Fragmentation (BPPIF) haves been introduced to model problems in diverse domains, like routing of consolidated traffic in optical networks and VLSI circuit design [1]. In their bin-minimization variant a set I of items, each having a size d_i, and a set of bins J, each having a capacity C, are given, together with a *fragmentation budget* F. The aim is to assign items to the minimum number of bins; up to F item splits are

[1] Email: `marco.casazza@upmc.fr`

[2] Email: `alberto.ceselli@unimi.it`, partially funded by Regione Lombardia - Fondazione Cariplo, grant n. 2015-0717, project "REDNEAT"

http://dx.doi.org/10.1016/j.endm.2016.10.016

allowed: whenever an item is split, it is replaced by two fragments; the split point is arbitrary, but the sum of fragment sizes must equal the size of the original item. Recursive fragmentations are allowed, but each split counts in the budget. The final set of fragments need then to be assigned to the bins, in such a way that the sum of item (fragment) sizes assigned to the same bin do not exceed C. Recent contributions to the field include both approximation algorithms [3] and exact methods [2], both approaches proving to be effective. As stressed in [3], major interest is currently in making BPPIF models more flexible. In this paper we tackle the generalization of BPPIF, in its bin-minimization variant, in which each bin $j \in J$ has a potentially different cost v_j and capacity c_j, and the overall cost of the used bins needs to be minimized. We refer to our generalization as the Variable Cost and Size BPPIF (VCSB).

2 Model

We first observe the following.

Proposition 2.1 *An optimal VCSB solution always exists, in which (a) each item is split in at most two fragments (b) each bin contains at most two fragmented items (c) each set of k bins contains at most $k-1$ fragmented items.*

Any solution satisfying (a)–(c) is called *primitive* [1]. A formal proof is omitted, but intuitively given a set of k bins and a solution assigning a subset of items $\bar{I} \subseteq I$ to them, a *Next Fit with Fragmentations* procedure produces a fragmentation pattern that comply with (a)–(c). Fragmented items link one bin another in a *chain* structure, that includes a subset $\bar{I} \subseteq I$ of items and a subset $\bar{J} \subseteq J$ of bins. On feasible chains it always holds $\sum_{i \in \bar{I}} d_i \leq \sum_{j \in \bar{J}} c_j$. Let Ω be the set of all feasible chains. Following the framework of [2] we model the VSCB with the following chain-based extended formulation:

$$\min \sum_{p \in \Omega} \Big(\sum_{j \in J} v_j \bar{y}_j^p\Big) z^p \tag{1}$$

$$\text{s.t.} \sum_{p \in \Omega} \bar{x}_i^p z^p = 1 \qquad \forall i \in I \tag{2}$$

$$\sum_{p \in \Omega} \bar{y}_j^p z^p \leq 1 \qquad \forall j \in J \tag{3}$$

$$\sum_{p \in \Omega} \bar{f}^p z^p \leq F \tag{4}$$

$$z^p \in \{0,1\} \qquad \forall p \in \Omega. \tag{5}$$

Coefficient $\bar{x}_i^p$ (resp. $\bar{y}_j^p$) is 1 if item i (resp. bin j) is included in chain p, 0 otherwise. Coefficient $\bar{f}^p$ is the number of fragmentations performed in chain p. Binary variables z^p are 1 if chain p is selected, 0 otherwise. Since $(\sum_{j\in J} v_j \bar{y}_j^p)$ represents the cost of using the set of bins in chain p, the objective function (1) aims at minimizing the overall cost of selected chains. Constraints (2) ensure that each item is included in a selected chain. Constraints (3) ensure that each bin is included in at most one selected chain. Constraints (4) enforce the fragmentations budget to be respected.

3 Algorithms

Formulation (1)–(5) includes an exponential number of variables. In order to obtain dual bounds on the value of the optimal solution we relax integrality conditions and exploit column generation techniques. Without loss of quality in the bound, we also relax constraints (2) in $\geq$ form. Let $\lambda_i \geq 0$, $\mu_j \leq 0$ and $\eta \leq 0$ be the dual variables associated to constraints (2), (3) and (4), resp.. The associated pricing problem is the following.

$$\begin{aligned}
\min & \sum_{j\in J}(v_j - \mu_j) - \sum_{i\in I}\lambda_i x_i - \eta f \\
\text{s.t.} & \sum_{i\in I} d_i x_i \leq \sum_{j\in J} c_j y_j \\
& \sum_{j\in J} y_j \leq f + 1 \\
& x_i \in \{0,1\} \quad \forall i \in I, \qquad y_j \in \{0,1\} \quad \forall j \in J, \qquad f \geq 0
\end{aligned}$$

Since if $\sum_{j\in J} y_j = 0$ also $\sum_{i\in I} x_i = 0$, such a setting is never profitable. Therefore we set $f = (\sum_{j\in J} y_j) - 1$, obtaining a variant of a 0–1 Knapsack Problem (KP) in which capacity consumption has a (possibly non monotone) cost. We solve it with an ad-hoc pseudo-polynomial time algorithm, whose main idea is to find, for each value of capacity $c = 0 \ldots \sum_{j\in j} c_j$ (a) the combination of bins of minimum reduced cost giving at least overall capacity c (b) the combination of items of minimum reduced cost using at most capacity c (c) sum up these two contributions to obtain an optimal pricing solution using capacity c (d) return the best pricing solution over all values of c. The key observation is that both steps (a) and (b) can be performed by solving *a single* KP each, that in turn can be done in pseudo-linear time. Therefore, our pricing algorithm has pseudo-linear time as well. When column generation is over we also perform rounding to search for good primal solutions.

Instances		CPLEX			CG		
Cap.	Weight	S	G(%)	T(s)	S	G(%)	T(s)
T	L	0	8.57	0.11	9	0.49	0.64
T	M	0	10.00	0.04	0	9.15	1.06
T	S	0	16.67	0.03	0	16.53	0.99
L	L	0	25.00	0.10	10	0.00	0.44
L	M	0	12.12	0.10	4	2.24	0.60
L	S	1	10.00	0.06	0	10.43	0.90
Overall		1	13.73	0.073	23	6.47	0.771

Table 1
Results on instances adapted from [1].

We implemented our algorithms in C++, using SCIP 3.1 as framework and CPLEX 12.6.2 to solve LP subproblems. Our tests ran on a PC with a 2.1GHz CPU and 8GB of RAM. We compared to the branch-and-cut algorithm of CPLEX, with default parameter settings, exploiting a compact formulation of the VSCBPP adapted from [2], and stopping the computation at the root node. We considered a dataset adapted from the literature [1]. The dataset includes instances with either Tight (T) or Loose (L) capacities, and items whose size is either Small (S), Medium (M) or Large (L). Preliminary results on 6 classes of 10 instances each with $|I| = 20$ are reported in Table 1. Capacity and size distribution are indicated in the first two colums. The Table includes two blocks, one for CPLEX and one for our Column Generation algorithm (CG). Each block reports the number of instances whose optimality was proved (S) the average optimality gap obtained (G) and the time required to complete the computation (T). CPLEX turned out to be faster, but CG results were more accurate, requiring reasonable additional CPU time. In particular, CG was able to directly solve many more instances to proven optimality.

References

[1] M. Casazza, A. Ceselli "Mathematical programming algorithms for bin packing problems with item fragmentation " Computers and Operations Research 46 (2014)

[2] M. Casazza, A. Ceselli "Exactly solving packing problems with fragmentation " Optimization Online Tech. Rep. (2015)

[3] B. LeCun, T. Mautor, F. Quessette, M.-A. Weisser "Bin packing with fragmentable items " Theoretical Computer Science 602 (2015)