ARTICLE IN PRESS

Omega ∎ (∎∎∎∎) ∎∎∎–∎∎∎

Contents lists available at ScienceDirect

Omega

journal homepage: www.elsevier.com/locate/omega

Maximum lateness minimization in one-dimensional bin packing ☆

Claudio Arbib [a], Fabrizio Marinelli [b,*]

[a] Dipartimento di Scienze/Ingegneria dell'Informazione e Matematica, Università degli Studi dell'Aquila, Via Vetoio, Coppito, I-67010 L'Aquila, Italy
[b] Dipartimento di Ingegneria dell'Informazione, Università Politecnica delle Marche, Via Brecce Bianche, I-60131 Ancona, Italy

ARTICLE INFO

Article history:
Received 17 June 2015
Accepted 8 June 2016

Keywords:
One-dimensional bin packing
Scheduling
Mixed Integer programming
Integer reformulation

ABSTRACT

In the One-dimensional Bin Packing problem (1-BP) items of different lengths must be assigned to a minimum number of bins of unit length. Regarding each item as a job that requires unit time and some resource amount, and each bin as the total (discrete) resource available per time unit, the 1-BP objective is the minimization of the makespan $C_{max} = \max_j\{C_j\}$. We here generalize the problem to the case in which each item j is due by some date d_j: our objective is to minimize a convex combination of C_{max} and $L_{max} = \max_j\{C_j - d_j\}$. For this problem we propose a time-indexed Mixed Integer Linear Programming formulation. The formulation can be decomposed and solved by column generation relegating single-bin packing to a pricing problem to be solved dynamically. We use bounds to (individual terms of) the objective function to address the oddity of activation constraints. In this way, we get very good gaps for instances that are considered difficult for the 1-BP.

1. Introduction

In Bin Packing, a set J of n items of distinct sizes must be assigned to a minimum number of identical bins, so that the size of the items assigned to any bin never exceed its capacity. In the (orthogonal) s-dimensional problem, items and bins are closed intervals of IR^s, and the former must be placed into the latter with no overlap. Items can or cannot be rotated before placement: in the latter case, the edge lengths of each interval can be normalized, and bins become unit s-cubes.

One can interpret the s-dimensional Bin Packing as a scheduling problem with n jobs of unit time length: when scheduled, job j consumes some fraction of a discretized resource, the bin, available in one unit per time unit. In general, applications include all those cases (e.g., ads scheduling in sponsored internet search [1]) in which the resource used has both a geometric and a time dimension. Here are other popular applications:

- in s-dimensional cutting, jobs are parts to be produced, and the resource is a stock of standard size from which smaller items must be cut [2, 5–7,14,17, 21, 31];
- in telecommunication channel scheduling, jobs are packets of known length, and the resource is a frame able to host packets up to a given total length [4,11].

Under common assumptions, completion times corresponds to stock positions in the sequence, and minimizing C_{max} means minimizing the number of resource units used: standard sizes in cutting problems, frames in packet scheduling, etc. But C_{max} is just one of the many scheduling objectives one can be interested in. To generalize, call C_j the completion time of j (that is: item j is assigned to the C_j-th bin) and associate j with a cost function $f_j(C_j)$. In multi-objective scheduling, a solution is evaluated through several functions $f_j^k(C_j)$, $k = 1, \ldots, R$. Often, a multi-objective is summarized by a convex combination of functions obtained from the $f_j^k(C_j)$:

$$f(C_1, \ldots, C_n) = \sum_{k=1}^{r} \alpha_k \max_{j \in J}\{f_j^k(C_j)\} + \sum_{k=r+1}^{R} \alpha_k \sum_{j \in J} f_j^k(C_j)$$

with $\sum_{k=1}^{R} \alpha_k = 1, \alpha_k \geq 0, k = 1, \ldots, R$.

If a function is non-decreasing with C_j, then it is called *regular* [20, chapter 2]. When item j is due by a specific date d_j, the following regular functions are frequently taken into consideration:

- *Tardiness*: $f_j(C_j) = T_j = \max\{C_j - d_j, 0\}$;
- *Lateness*: $f_j(C_j) = L_j = T_j - E_j = C_j - d_j$.

☆This manuscript was processed by Associate Editor Kis.

* Corresponding author.
E-mail addresses: claudio.arbib@univaq.it (C. Arbib), fabrizio.marinelli@univpm.it (F. Marinelli).

http://dx.doi.org/10.1016/j.omega.2016.06.003

1.1. Our problem

The general function $f(C_1, \ldots, C_n)$ combines min–max and min-sum terms. In this paper we focus on a pure min–max problem with $r = R = 2$:

$$f_j^1(C_j) = C_j \quad f_j^2(C_j) = C_j - d_j$$

The second function is the lateness of item j: namely, we seek for a pack-and-schedule that solves

$$\min_{C_1,\ldots,C_n} f(C_1, \ldots, C_n) = \alpha_1 C_{max} + \alpha_2 L_{max} \tag{1}$$

for given rational constants $\alpha_1, \alpha_2 \geq 0$ such that $\alpha_1 + \alpha_2 = 1$. The opportunity of giving different weights to material and lateness costs is much application-dependent. There are relevant industrial cases in which material cost is closely comparable to, and sometimes larger than, the cost of delay (see e.g. [3]). In any case, α_1 and α_2 derive from the real costs of bin usage and time. When these costs cannot be easily evaluated, it is more appropriate to keep separate goals and transform one or both terms of the objective function into constraints (if both, we face a feasibility problem and speak of *goal-programming*). Our model naturally fits with this approach, see Section 2.4.

Models and methods will be developed according to the following

Assumption 1.1. In the definition of problem (1), we assume:

(i) *constant cut time* (this common assumption, see [2,7,21,26,31], may however be not obvious, especially for $s > 1$: as item dimension increases, the time for item placement in – or cut from – a bin may change very much from pattern to pattern);
(ii) *due-dates integer multiple of cut time* (irrelevant for other cost functions, such as tardy jobs, but generally not irrelevant for lateness).

1.2. Literature review

Scheduling objectives in cutting and packing problems are receiving increasing attention. Among many papers concerned on cutting (see bibliography), [2,7,21] are the most recent and the closest to our situation:

- Reference [21] proposes an integer programming based heuristic to minimize a combination of trim-loss ($= C_{max}$) and total weighted tardiness.
- Reference [2] addresses the same problem as [21] by exact models, either with or without column generation. The model has variables associated with time-periods: in order to limit the number of variables, period lengths are adjusted by an ad-hoc procedure.
- Reference [7] develops a genetic heuristic for 2-dimensional, non-oriented, single bin size trying to approximate the Pareto frontier for the criteria of bin and maximum lateness minimization: this is the problem considered by us, although our computational experience is limited to 1-dimensional packing.

Parallel machine scheduling is a classical counterpart of bin packing: instead of being minimized, bins are given and the typical objective is to minimize the makespan (intended as the maximum load of a bin). Indeed, bin packing and parallel machines scheduling can be seen as "orthogonal" special cases of Cumulative Resource Scheduling [16], a problem in which each job consumes some amount of a shared resource up to availability.

A more general additive criterion is considered in [9,27–29], where precedence or time-indexed formulations are developed and decomposed in order to solve the problem by column generation:

- Reference [9] formulates $P \| \sum f_j(C_j)$ using decision variables that describe precedence relations among jobs on any machine; the master problem derived from reformulation is in the shape of Set Partitioning.
- Reference [27] focuses on $P \| \sum w_j C_j$ and directly formulates the decomposed problem with variables associated to feasible machine schedules. See also [28].
- Reference [29] assumes the general criterion $\sum f_j(C_j)$, and – as in our case – decomposition is applied to a time-indexed model. Unlike our case, however, jobs have non-unit processing times and do not consume other resource but time: the problem has therefore a special structure (interval matrix).

The time-indexed approaches listed above are very close to ours: the main difference is that we do not deal with a parallel-machine setting, therefore Dantzig–Wolfe decomposition is applied to different formulations.

A final note on complexity: $P \| (\alpha_1 C_{max} + \alpha_2 L_{max})$ is NP-hard for two machines and any values of $\alpha_1, \alpha_2 \geq 0$ with $\alpha_1 + \alpha_2 = 1$; when due dates are identical, the problem becomes Multiprocessor Scheduling. On the other hand, $1 \| C_{max}$ is trivial and $1 \| L_{max}$ can be solved in $n \log(n)$ time by Early Due Date priority rule (EDD). Whatever are the due dates, a schedule minimizing L_{max} is necessarily *active* [20, ch. 2], hence its C_{max} always equals the sum of processing times. Thus, $1 \| (\alpha_1 C_{max} + \alpha_2 L_{max})$ has the same optimal solution as $1 \| L_{max}$. For a comprehensive discussion on multicriteria scheduling problems see [25].

1.3. This contribution

Since bin packing is equivalent to cutting stock with unit demand, one can tackle due dates by a formulation of the cutting stock problem as in [2]. The general method is close to that applied to parallel scheduling in [27,28]. The objective here considered replaces however total tardiness with maximum lateness (see also [7]): with time-indexed formulations a min-sum term can in fact be less problematic because, unlike min-max, does not need activation constraints.

In this paper we show how to obtain guaranteed approximation algorithms for this problem from existing guaranteed approximation algorithms for Bin Packing. We also propose exact Mixed Integer Linear Programs based on time-indexing, and improve them by a careful use of lower and upper bounds so as to solve difficult problem instances. The main dataset for the numerical experiments was constructed on non-IRUP bin packing problems from the literature: with our approach, we were able to solve in few seconds problems with up to 120 parts and, for problems with 200 parts, reach very small gaps (less than 2%) in acceptable time (less than 600 s).

Formulations and an approximation result are detailed in Section 2. We investigate ways to improve the formulation so as to address quite large problem instances: a key issue to achieve efficiency is to take advantage of lower and upper bounds to C_{max}, L_{max} and to the global objective function (1). The bounds and their implementation in the formulations are discussed in Section 3. A computational experience based on [12] and mainly focussed on $\alpha_1 = \alpha_2$ proves the validity of the method, and is reported in Section 4. Conclusions and directions for future research are drawn in Section 5.