New Bin Packing Fast Lower Bounds

New Bin Packing Fast Lower Bounds

Teodor Gabriel Crainic
Département de management et technologie, U.Q.A.M.
and
Centre de recherche sur les transports, U. de Montréal
C.P. 6128, Succursale Centre-ville - H3C 3J7 Montral (QC) Canada

Guido Perboli *
Department of Control and Computer Engineering
Politecnico di Torino
Corso Duca degli Abruzzi, 24 - I-10129 Torino (Italy)
Tel: +39 011 5647097
Fax: +39 011 5647099
e-mail: guido.perboli@polito.it

Miriam Pezzuto
Department of Control and Computer Engineering
Politecnico di Torino
Corso Duca degli Abruzzi, 24 - I-10129 Torino (Italy)

Roberto Tadei
Department of Control and Computer Engineering
Politecnico di Torino
Corso Duca degli Abruzzi, 24 - I-10129 Torino (Italy)

* Corresponding Author

Abstract

In this paper, we address the issue of computing fast lower bounds for the Bin Packing problem, i.e., bounds that have a computational complexity dominated by the complexity of ordering the items by non-increasing values of their volume. We introduce new classes of fast lower bounds with improved asymptotic worst-case performance compared to well-known results for similar computational effort. Experimental results on a large set of problem instances indicate that the proposed bounds reduce both the deviation from the optimum and the computational effort.
Keywords: Bin Packing, lower bounds

1 Introduction

Given a set of n items i, with volumes v_i, $i \in 1, ..., n$, and containers of fixed capacity V called bins, the *Bin Packing* (BP) problem consists in assigning each item to a bin, such that: 1) the sum of the volumes of the items within the same bin is not greater than the capacity of the bin; 2) the number of used bins is minimal. The BP problem is known to be NP-hard [10].

The BP problem occurs as sub-problem in many other settings, including multi-dimensional bin packing, strip packing, network design, circuit design, and loading in flexible manufacturing systems. An important area of research is thus dedicated to the development of bounding procedures that yield tight bounds with limited computational effort. *Fast Lower Bounds* ($BPFB$s), i.e., bounds that have a computational complexity dominated by the complexity of ordering the items by non-increasing values of their volume, belong to this category.

The goal and main contribution of this paper is to introduce Fast Lower Bounds able to extend and outperform existing Fast Lower Bounds. The performance of the new bounds is analyzed both theoretically, by determining their asymptotic worst-case performance, and computationally by means of an extensive set of tests.

It is indeed known that both measures are required to correctly characterize the behavior of lower bounds. Worst-case asymptotic analysis provides an understanding of this behavior when the size of the problem instances increases. But this analysis is not sufficient to completely characterize the performance of a lower bound. Thus, for example, lower bounds for the BP problem with worst-case ratio asymptotically approximated to 1 can theoretically be obtained in polynomial time, but the process cannot be applied in practice for computational reasons (see [5] for further details). Moreover, lower bounds with very different results in quality and computational effort display the same asymptotic behavior [7].

The new Fast Lower Bounds are able to achieve an asymptotic worst-case performance of at least 3/4 independently on the Fast Lower Bound they are extending.

Moreover, experimental results indicate that one of the proposed lower bound achieves the best results when compared to the other Fast Lower

Bounds, reducing both the deviation of the bound from the optimum and the computational effort by a significant factor. These results are obtained in particular for some hard-to-solve instances.

The paper is organized as follows. We recall existing lower bounds for the BP problem in Section 2. Section 3 is dedicated to the introduction of two new $BPFB$s. For each of them, we discuss the asymptotic worst-case analysis in Section 4 and present the computational performance in Section 5. The Annex contains the pseudo-code description of the various procedures presented in the paper.

2 Existing Lower Bounds for the BP Problem

Let us assume, without loss of generality, that the volume of item i, v_i, $i \in 1, ..., n$, is at most equal to the bin capacity V. One may then normalize the volumes in the interval $[0, 1]$, where 1 corresponds to the bin capacity V. Define the following set of variables:

x_{ij} : binary variable that is equal to 1 if item i is assigned to bin j, 0 otherwise;

y_j : binary variable that is equal to 1 if bin j contains at least one item, 0 otherwise.

The BP problem can then be formulated as follows:

$$min \quad \sum_{j=1}^{n} y_j \tag{1}$$

$$\sum_{i=1}^{n} v_i x_{ij} \leq y_j \quad j = \{1, ..., n\} \tag{2}$$

$$\sum_{j=1}^{n} x_{ij} = 1 \quad i = \{1, ..., n\} \tag{3}$$

$$y_j \in \{0, 1\}, \forall j \tag{4}$$

$$x_{ij} \in \{0, 1\}, \forall i, \forall j. \tag{5}$$

The objective function (1) minimizes the number of used bins. Constraints (2) enforce the bin capacity restrictions and force variables y_j to assume value 1 if at least one item i is assigned to bin j. Equations (3) force to load each item into just one bin. In the following, we present existing lower bounds and discuss their performance. We first introduce existing $BPFB$s (Sub-section 2.1). We then present lower bounds characterized by heavier computational complexity (Sub-section 2.2).

2.1 Fast Lower Bounds for the BP Problem

Elementary bounds can be derived from the continuous, Lagrangian, and surrogate relaxations of the model (1) - (5).

Martello and Toth [15] proved that bounds obtained from the continuous and surrogate relaxations are dominated by the lower bound

$$L_1(I) = \left\lceil \sum_{i=1}^{n} v_i \right\rceil. \tag{6}$$

The bound L_1 can be computed in $O(n)$ and it has a worst-case performance of $\frac{1}{2}$ [15, 16]. The bound obtained from the Lagrangian relaxation of constraints (2) is also dominated by L_1 [15].

A tighter bound was obtained by focusing on the items with $v_i > \frac{1}{2}$ [15, 16]. The main idea is that, given a value $\varepsilon \in \left[0, \frac{1}{2}\right]$, the items with a volume $v_i > 1-\varepsilon$ cannot be loaded together. Moreover, the authors suppose that the items with volume $v_i < \varepsilon$ are used to completely fill the bins in which the items with volume $v_i > 1-\varepsilon$ are loaded. For uniformity of notation, we report in the following the formulation of the bound given by Fekete and Schepers in [9]:

$$\begin{aligned} L_2(I) &= \max_{\epsilon \in \left[0,\frac{1}{2}\right]} \left(L_2(I,\epsilon)\right), \\ L_2(I,\epsilon) &= \left|\{i \in I \mid v_i > 1-\epsilon\}\right| + L_1\left(\{i \in I \mid \epsilon \le v_i \le 1-\epsilon\}\right). \end{aligned} \tag{7}$$

In [13] Haouaria and Gharbia show that in the formulation by Fekete and Schepers the only relevant values of ϵ are $\epsilon \in V'$, where $V' = \{v_i \in [0, 1/2]\} \cup \{1/2\}$.

The worst-case performance of L_2 is $\frac{2}{3}$ and it can be computed in $O(n)$ if the items are sorted by non-increasing volume.

Labbé, Laporte, and Mercure improved L_2 [14]. Let us define $I(a,b) = \{i \in I | a < v_i \le b\}$. Their bound L_{LLM} focuses on the items in $I(\frac{1}{3}, \frac{1}{2})$ that cannot be loaded together with the items in $I(\frac{1}{2}, 1)$.

The authors make use of an adapted First Fit Procedure [6] that first assigns each item in $I(\frac{1}{2}, 1)$ to a different bin. The procedure then assigns items with the smallest volumes to the bins with the smallest residual capacities. Let I_3 be the set of items with volumes $1/3 < v_i \le 1/2$ which cannot be assigned to a bin according to this procedure. Thus, they require at least $\lceil |I_3|/2 \rceil$ additional bins. Using the formalization given by Bourjolly and Rebetez [3], the bound L_{LLM} can be expressed as:

$$\begin{aligned} L_{LLM}(I) &= \max_{\epsilon \in \left[0,\frac{1}{2}\right]} \{L_{LLM}(I,\epsilon)\} \\ L_{LLM}(I,\epsilon) &= |I(1/2,1)| + \left\lceil \frac{|I_3|}{2} \right\rceil + p(\epsilon) \\ p(\epsilon) &= \max\left(0, \left\lceil \sum_{\{\epsilon \le v_i \le 1-\epsilon\}} v_i - \left| I\left(\frac{1}{2}, 1-\epsilon\right)\right| - \left\lceil \frac{|I_3|}{2} \right\rceil \right\rceil \right). \end{aligned} \tag{8}$$

Bourjolly and Rebetez [3] proved that this bound has an asymptotic worst-case ratio of $\frac{3}{4}$. L_{LLM} can be computed in time $O(n \lg n)$ if the items are unsorted and in $O(n)$ if the items are sorted by non-increasing volumes.

A new class of $BPFB$s has been introduced by Fekete and Schepers [9]. It is based on the use of item volumes modified by Dual Feasible Functions. A function $u : [0,1] \longrightarrow [0,1]$ is called dual feasible if, given a finite set F such that $\sum_{x \in F} x \leq 1$, then $\sum_{x \in F} u(x) \leq 1$. The authors also proved that, given a lower bound of the BP problem, new lower bounds can be obtained by applying the same lower bound on the volumes modified by one or more dual feasible functions. Their bound, called $L_{FS}^{*(p)}$, is defined as follows:

$$L_{FS}^{*(p)} = \max\left\{L_2(I), max_{k=2,\ldots,p} L_{FS}^{(k)}(I)\right\},$$

where

$$L_{FS}^{(k)}(I) = \max_{\epsilon \in \left[0, \frac{1}{2}\right]} L_1\left(u^{(k)} \circ U^{(\epsilon)}(I)\right)$$

$$u^{(k)}(x) : [0,1] \to [0,1], u^{(k)}(x) = \begin{cases} x & x(k+1) \in \mathbb{N} \\ \frac{\lfloor x(k+1) \rfloor}{k} & \text{otherwise} \end{cases}$$

$$U^{(\epsilon)}(x) : [0,1] \to [0,1], U^{(\epsilon)}(x) = \begin{cases} 1 & x > 1 - \epsilon \\ x & \epsilon \leq x \leq 1 - \epsilon \\ 0 & x < \epsilon. \end{cases}$$

The dual feasible function $u^{(k)}$ maps, for a given value k, the volumes up to $2k+1$ distinct values, while $U^{(\epsilon)}$ is the expression (7) written as dual feasible function.

Fekete and Schepers proved that $L_{FS}^{*(p)}$ has an asymptotic worst-case performance of $\frac{3}{4}$ for $p \geq 2$ and can be computed in $O(pn)$ if the items are sorted by non-increasing volumes. Considering that p is a constant not related to the problem instance but fixed by the user, this bound can be considered $O(n)$ when p is sufficiently small (e.g., $p \leq 100$). Moreover, extensive computational results show that $p = 20$ provides the best tradeoff between computational efficiency and quality of solution.

2.2 Other Lower Bounds for the BP Problem

One of the tightest bounds for the BP problem, known as L_3, has been developed by Martello and Toth [15]. The authors use a reduction procedure called $MTRP$ to build the non-dominated bins in an optimal solution that contains at most 3 items (if such bins exist). The corresponding items are then removed and the L_2 bound is applied on the remaining ones.

Consider an instance I of the BP problem where the items are sorted by non-increasing volumes and define a *Feasible Set* (FS) as a set of items for which the sum of volumes is not greater than the capacity of a bin. Then, for a given item i, the maximum cardinality of the FSs that include i in an optimal solution is

$$F_{i,\max} = 1 + \begin{cases} 0 & v_i + v_N > 1 \\ \max_{k \in I \setminus \{i\}} \left\{k : \sum_{j=1}^{k} v_{N-j+1} \leq 1 - v_i\right\} & \text{otherwise.} \end{cases}$$

The $MTRP$ procedure is then based on the following dominance criteria:

D1 If $F_{1,\max} = 1$, then, there exists an optimal solution containing the FS $F = \{v_1\}$.

D2 If $F_{1,\max} = 2$ or $\exists \overline{k} : (v_1 + v_{\overline{k}}) = 1$, then, there exists an optimal solution containing the FS $F = \left\{1, \overline{j}\right\}$, where

$$\overline{j} = \begin{cases} \arg\min_{j:\{1,j\} \text{ is a FS}} \{1 - (v_1 + v_j)\} & \text{if } F_{1,\max} = 2 \\ \overline{k} & \text{if } (v_1 + v_{\overline{k}}) = 1. \end{cases}$$

D3 If $F_{1,\max} = 3$, then, there exists an optimal solution containing the feasible set:

(a) $F = \left\{1, \overline{k}\right\}$, $\overline{k} = \arg\min_{\{1,k\} \text{ is a FS}} \{1 - (v_1 + v_k)\}$ if $\nexists \{i, j\} \in I$: $1 - (v_1 + v_i + v_j) \leq 1 - (v_1 + v_k)$, $\{1, i, j\}$ is a FS;

(b) $F = \left\{1, \overline{i}, \overline{j}\right\}$, $\left\{\overline{i}, \overline{j}\right\} = \arg\min_{\{1,i,j\} \text{ is a FS}, i<j} \{1 - (v_1 + v_i + v_j)\}$, if $\overline{i} = \arg\min_{\{1,i\} \text{ is a FS}} \{1 - (v_1 + v_i)\}$ and $i - j \leq 2$;

(c) $F = \left\{1, \overline{i}, \overline{j}\right\}, \left\{\overline{i}, \overline{j}\right\} = \arg\min_{\{1,i,j\} \text{ is a FS}, i<j} \{1 - (v_1 + v_i + v_j)\}$, if $\overline{i} = \arg\min_{\{1,i\} \text{ is a FS}} \{1 - (v_1 + v_i)\}$ and $v_1 + v_{j-1} + v_{j-2} > 1$.

The $MTRP$ procedure can be applied in $O(n^2)$ and L_3 in $O(n^3)$. The bound L_3 is quite time consuming. However, computational experiments show that this bound is very tight [15]. It has been proved recently that its asymptotic worst-case ratio is 3/4 [7, 17].

An alternative formulation of the BP problem is given by Gilmore and Gomory [11, 12]. In this formulation, an integer variable λ_q is introduced for each FS of the instance. The variable represents the number of times FS q is used in the optimal solution. Let Q be the set of the FSs and q_i a constant that is equal to 1 if item i is contained in FS q, 0 otherwise. Then, the IP formulation is as follows

$$min \quad \sum_{q \in Q} \lambda_q \tag{9}$$

$$\sum_{q \in Q} q_i \lambda_q = 1 \quad i = \{1, ..., n\} \tag{10}$$

$$\lambda_q \in N^+. \tag{11}$$

The lower bound L_{cg} derived by relaxing constraints (11) of model (9) - (11) dominates the continuous bound L_1 [11, 12]. Due to the large number of columns involved, Vanderbeck solved the relaxed model by means of a column generation method [20]. This bound obtained better results than L_3 on some hard instances.

Chen and Srivastava [5] introduced a family of lower bounds that can be expressed as follows :

$$L_{CS}(I, m) = \max \{Opt(1/m, 1), L_1(I)\}, \tag{12}$$

where $Opt(1/m, 1)$ is the optimal solution of the instance obtained considering the items in I with volumes greater than $1/m$. The authors derived

$BPFB$s for $m \leq 3$. Moreover, the authors proved that the worst-case performance of their bound is 2/3 for $m \geq 2$, while the bounds have an asymptotic worst-case ratio of $m/m+1$. The complexity of the bounds is dominated by the complexity of computing $Opt(1/m, 1)$, which increases with the value of m.

3 New Fast Lower Bounds

$BPFB$s are a sub-set of BP lower bounds characterized by a computational effort that does not exceed the computational effort of ordering items by non-increasing volumes. These bounds are important because many problem classes include the BP problem as sub-problem and require the computation of a good approximation of its optimum with a small computational effort. We introduce two general classes of $BPFB$s that yield the current best fast bounding procedure according to both solution quality and computational effort (see Sections 4 and 5).

3.1 Truncated-reduction lower bounds

Consider an instance on the BP problem. The basic idea is to first reduce the instance using the dominance criteria D1 and D2 and, then, to apply a fast lower bound to the instance with the remaining items. The reduction procedure is called TMTRP (Algorithm 1 in the Annex). TMTRP proceeds iteratively and considers one item at a time. If the dominance criterion D1 or D2 holds, the non-dominated FS is built and the corresponding item is removed from the original instance. Assuming items are sorted by non-increasing volumes, each item is considered at most twice. Dominance can be verified in $O(n)$ time and, thus, the TMTRP procedure is $O(n)$ if the items are in non-increasing order of volume (for further details, see [17]). Moreover, according to the following theorem, the procedure TMTRP solves to optimality all the instances of the BP problem where the smallest item has a volume greater than 1/3.

Theorem 1 *Given an instance I of the BP problem with all the volumes $v_i > \frac{1}{3}$, the procedure TMTRP finds an optimal solution of the instance.*

Proof. Order the items by non-increasing values of the volumes v_i and consider the first item of the instance. Then, $v_1 = \max_i \{v_i\}$. We have to check the following cases:

1. $|I| = 1$. An optimal solution is 1.
2. $v_1 \in \left(\frac{2}{3}, 1\right]$. We know (hypothesis) that $v_N > \frac{1}{3}$, so $v_1 + v_N > 1$. By dominance criterion D1, the FS $F_1 = \{1\}$ is in an optimal solution, thus

$$OPT(I) = 1 + OPT(I')$$
$$I' = I \backslash F_1.$$

3. $v_1 \in \left(\frac{1}{3}, \frac{2}{3}\right]$ and $|I| = 2$. The procedure TMTRP returns an optimal value of 1.

4. $v_1 \in \left(\frac{1}{3}, \frac{2}{3}\right]$ and $|I| \geq 3$. If $v_1 + v_N > 1$, then D1 applies and the FS $F_1 = \{1\}$ is in an optimal solution.

   Otherwise, by hypothesis, we know that $v_i > \frac{1}{3}, \forall i$, so the following inequalities are satisfied

   $$(v_1 + v_N) \leq 1$$
   $$1 - (v_1 + v_N + v_{N-1}) < 0.$$

   By dominance criterion D2, we know that an item j exists such that the FS $F_1 = \{1, j\}$ is in an optimal solution.

   Thus,

   $$OPT\,(I) = 1 + OPT\,(I')$$
   $$I' = I \backslash F_1.$$

Consider now the instance I'. We know that either the hypotheses of the theorem are satisfied or the instance is empty. If the instance is not empty, we can apply recursively the cases 1-4 to the first item of the instance I', building a new non-dominated FS F_2. We continue with this process until the instance is empty. At the end of this process, we have that

$$OPT\,(I) = 1 + OPT\,(I') = 1 + 1 + OPT\,(I'') = \sum_{j=1}^{K} 1 + OPT\,(\{\emptyset\}),$$

where K is the number of sub-instances on which the cases 1-4 have been applied. Thus, the set I is split into the FSs F_i and each FS F_i is non-dominated. Then the solution

$$F_1, ..., F_K$$

is an optimal solution of the instance I and $OPT\,(I) = K$ is the optimal value. ■

New fast bounds, called *Truncated-Reduction Lower Bounds* (TRLB), can therefore be obtained as

$$LB_{TRLB} = \max\{B_{TMTRP}(I(1/3, 1)), B_{TMTRP} + FLB\,(TMTRP)\}, \quad (13)$$

where B_{TMTRP} are the bins loaded by the TMTRP procedure applied to I, $B_{TMTRP}(I(1/3, 1))$ are the bins loaded by the TMTRP procedure applied to the subset of I with volumes greater than 1/3, while $FLB(TMTRP)$ is a fast lower bound with $FLB\,(I) \geq L_1\,(I)$, applied to the instance with the remaining items.

Replacing FLB with L_2, $L_{FS}^{*(p)}$, and L_{LLM}, respectively, we obtain the bounds

$$LB_{TRLB_2} = \max\{B_{TMTRP}(I(1/3, 1)), B_{TMTRP} + L_2\,(TMTRP)\},$$
$$LB_{TRLB_{FS}}^{(p)} = \max\{B_{TMTRP}(I(1/3, 1)), B_{TMTRP} + L_{FS}^{*(p)}\,(TMTRP)\},$$
$$LB_{TRLB_{LLM}} = \max\{B_{TMTRP}(I(1/3, 1)), B_{TMTRP} + L_{LLM}\,(TMTRP)\}.$$

Given that L_2 is dominated by $L_{FS}^{*(p)}$, $LB_{TRLB_{FS}}^{(p)}$ dominates LB_{TRLB_2}. Both the TMTRP procedure and the lower bounds L_2, $L_{FS}^{*(p)}$, and L_{LLM} are $O(n)$ when the items are sorted by non-increasing volumes. Then, the lower bounds LB_{TRLB_2}, $LB_{TRLB_{FS}}^{(p)}$, and $LB_{TRLB_{LLM}}$ are $O(n)$ and can be classified as $BPFB$s.

The bounds LB_{TRLB_2}, $LB_{TRLB_{FS}}^{(p)}$, and $LB_{TRLB_{LLM}}$ dominate the bounds $L_{CS}(m), m \leq 3$. In fact, setting $m = 1$, the bound $L_{CS}(1)$ reduces to L_1, while setting $m = 2$ we obtain

$$L_{CS}(I, 2) = \max\{L_2(I, 1/2), L_1(I)\} \leq L_2(I)$$

and both $L_1(I)$ and $L_2(I)$ are dominated by LB_{TRLB_2}, $LB_{TRLB_{FS}}^{(p)}$ and $LB_{TRLB_{LLM}}$. For $m = 3$, we obtain

$$\begin{aligned} L_{CS}(3) = \quad & \max\{Opt(1/3, 1), L_1(I)\} = \max\{B_{TMTRP}(I(1/3, 1)), L_1(I)\} \\ & \leq \max\{B_{TMTRP}(I(1/3, 1)), B_{TMTRP} + L_1(TMTRP)\}, \end{aligned}$$

and $\max\{B_{TMTRP}(I(1/3, 1)), B_{TMTRP}+L_1(TMTRP)\}$ is dominated by LB_{TRLB_2}, $LB_{TRLB_{FS}}^{(p)}$, and $LB_{TRLB_{LLM}}$.

3.2 Dual feasible functions and reduction lower bounds

Computational experiments indicate that the bound L_3 is stronger than all currently known classes of Fast Lower Bounds [9, 15]. This is mainly due to the iterative use of the procedure $MTRP$ (see [15] for a description of the algorithm). Unfortunately, this bound has a computational complexity of $O(n^3)$. We present a new bound that builds on ideas from L_3, but reduces the overall complexity to $O(n)$.

Consider an instance I and suppose that the item volumes assume at most q different values, with q fixed. We can then derive a quicker version of the procedure $MTRP$, called CPPT_RED (Algorithm 2 in the Annex), by using a list of the different values of the volumes and the associated number of items with the same volume. We refer to each pair (volume value, number of items with this volume) as a *volume class*. Suppose there exist three procedures $FINDF1$, $FINDF2$, and $FINDF3$ (Algorithms 3, 4, and 5 in the Annex) that, given the list of the volume classes $lclass$ and the last volume class considered, build the non-dominated FSs that verify the dominance criteria D1, D2, and D3, respectively. More precisely, $FINDFi$, $i = 1, 2, 3$, builds the non-dominated FSs that verify the dominance criterion i = D1, D2, or D3. If $FINDFi$ verifies that the corresponding dominance criterion holds for the $k-tuple$ of items $(i_1, ..., i_k)$, then the criterion also holds for the $k-tuples$ $(i'_1, ..., i'_k)$ with the same item volume (i.e., $v_{i_1} = v_{i'_1}, ..., v_{i_k} = v_{i'_k}$). Finding these $k - tuples$ $(i'_1, ..., i'_k)$ can be accomplished in constant time by means of a list that associates the item volume values to volume classes. Then, the algorithm builds $m + 1$ FSs at each iteration, where m is the number of the $k - tuples$ $(i'_1, ..., i'_k)$.

According to the results of Fekete and Schepers, the bound L_3 applied to an instance modified by $u^{(k)}(x)$ is a bound of the original problem. Furthermore, the dual feasible function $u^{(k)}(x)$, given k, modifies the values of the

volumes of the items such that the volumes assume $2k+1$ different values only ($v_i = \frac{i}{k}, i \in \{0, ..., k\}$, and $v_i = \frac{j}{k+1}, j \in \{1, ..., k\}$). Thus, given a fixed value p, we can derive the following new bound

$$LB^{(k)}_{CPPT\_RED} = B_{CPPT\_RED}(I(u^k)) + LB^{(k)}_{FS} \tag{14}$$

$$LB^{(p)}_{DFFR} = \max_{k=1,...,p} \left\{ LB^{(k)}_{CPPT\_RED} \right\}, \tag{15}$$

where $B_{CPPT\_RED}(I(u^k))$ are the non-dominated FSs found by the $CPPT\_RED$ procedure applied to the instance where the volumes have been modified by the $u^{(k)}(x)$ dual feasible function and $LB^{(k)}_{FS}$ is applied to the remaining items. Using (15) as FLB in (13) yields

$$\begin{aligned} LB^{*(p)}_{DFFR} = &\max\{B_{TMTRP}(I(1/3, 1)), \\ &B_{TMTRP} + \max\left\{ LB^{(p)}_{DFFR}, LB^{*(p)}_{FS} \right\}\}. \end{aligned} \tag{16}$$

Both $LB^{*(p)}_{FS}$ and B_{TMTRP} are $O(n)$. Then, to prove that $LB^{*(p)}_{DFFR}$ is a $BPFB$, we have to prove that, given a fixed k, the $CPPT\_RED$ procedure is $O(n)$ too.

Theorem 2 *Given an instance I of the BP problem with at most q different values of the item volumes v_i, $q << n$, the complexity of the procedure CPPT_RED is $O(n)$.*

Proof. Let *lclass* be a list that associates each volume value to a volume class. *lclass* is built in $O(n)$ and can be ordered by non-increasing values of the volumes in $O(q \lg q)$. $FINDF1$ and $FINDF2$ have complexity $O(q)$, while $FINDF3$ is $O(q^2)$ because, in the worst case, one must test all the pairs j_1, $j_2 \in lclass$. Volume classes are examined successively. If $FINDF1$, $FINDF2$, and $FINDF3$ fail to build non-dominated FSs for a given volume class j, it is removed from *lclass* and one proceeds to the next one.

If the dominance criterion D1 holds for an item i in the volume class j, then it holds for all the items in j (by definition, the items in the same volume class have the same volume). Thus, $D1$ holds for all the items in j and $FINDF1$ can produce $|j|$ FSs. Moreover, the volume class j becomes empty and is removed from *lclass*.

Similarly, if dominance criterion D2 holds, then $FINDF2$ identifies a pair of items (i_1, i_2) that produces a non-dominated FS. Suppose, without loss of generality, that j is the volume class of i_1 and j' the volume class of i_2. If $j \neq j'$, we can build $min\{|j|, |j'|\}$ FSs and one of the two volume classes becomes empty (i.e., contains no more items) and is removed. Otherwise, $j = j'$ and we can build $\lfloor |j|/2 \rfloor$ FSs. If $|j|$ is even, once the FSs are built, the volume class is empty (all items with $v_i = j$ have been used if $|j|$ is even). Otherwise, if $|j|$ is odd, after building the FSs the volume class contains exactly one item.

Let us suppose that $FINDF1$ and $FINDF2$ fail to build FSs, but a triplet exists such that the dominance criterion D3 holds. We then consider the volume classes of the items in the triplet and, by an argument similar to

those used above for $FINDF2$, at least one of the volume classes becomes empty or holds at most two items.

If, after applying procedures $FINDF2$ and $FINDF3$, one or more FSs have been built but no volume class became empty, then at least one volume class contains one or two items only. When procedure CPPT_RED considers this volume class again, by applying $FINDF1$, $FINDF2$ or $FINDF3$, the volume class becomes empty if it contains one item. Otherwise, if the volume class contains two items, after reconsideration it will contain either 1 item (and it will be emptied if it will be considered for the third time) or it will be empty. Consequently, each volume class is considered at most three times and the procedure performs at most $3q\left(q+q+q^2\right)$ operations. Because $FINDF1$, $FINDF2$, and $FINDF3$ are applied at most q times, the procedure CPPT_RED has an $O\left(n+q*3q\left(qn+qn+q^2n\right)\right) = O\left(n+q^3\right)$ complexity. With q fixed and sufficiently small, the procedure CPPT_RED has complexity $O\left(n\right)$. ■

L_1 and L_2 are dominated by $LB_{FS}^{*(p)}$, $L_{CS}(1) = L_1$, and $L_{CS}(2)$ is dominated by L_2. Then, the bound $LB_{DFFR}^{*(p)}$ dominates L_1, L_2, $LB_{FS}^{*(p)}$, $L_{CS}(1)$, and $L_{CS}(2)$. Moreover, deriving $LB_{DFFR}^{*(p)}$ from (13), we obtain:

$$\begin{aligned} L_{CS}(3) =& \max\left\{Opt(1/m,1), L_1(I)\right\} = \max\left\{B_{TMTRP}(I(1/3,1)), L_1(I)\right\} \\ \leq& \max\{B_{TMTRP}(I(1/3,1)), B_{TMTRP} + L_1\left(TMTRP\right)\} \\ \leq& LB_{DFFR}^{*(p)}. \end{aligned}$$

4 Asymptotic Worst-Case Analysis of the New Fast Lower Bounds

The asymptotic worst-case analysis provides a better understanding of the behavior of a lower bound when the size of the problem instances is increasing. It is a well known and widely used theoretical performance measure that complements an experimental performance evaluation.

Let I be an instance of the BP problem and L a lower bound on the optimum value $OPT\left(I\right)$. The performance of L for the instance I is defined as $\frac{L(I)}{OPT(I)}$. The asymptotic worst-case analysis of the performance of L is given by:

Definition 3 (Asymptotic worst-case) *Let I be a generic BP instance, L a lower bound for the BP problem, and $OPT\left(I\right)$ the optimum of I. Given a positive integer s, the* asymptotic worst-case *of L is given by*

$$r_\infty\left(L\right) = lim_{s\to\infty} sup\left\{\frac{L\left(I\right)}{OPT\left(I\right)} \mid \forall I \text{ with } OPT\left(I\right) \geq s\right\}.$$

It is known that lower bounds of the BP problem with an asymptotic worst case converging to 1 can be defined (e.g, the family of lower bounds defined by Chen and Srivastava [5]). Performing the asymptotic worst-case analysis is, generally, a hard task, however. The authors proved in [7] and [17] the following general result that reduces the complexity of computing the asymptotic worst-case of BP lower bounds.

Theorem 4 *Given a value $k > 1, k \in \mathbb{N}$ and a lower bound LB of the BP problem such that*

$$\begin{aligned} &H1)\ LB\left(I'\right) = OPT\left(I'\right), \forall\, I' : \min_{i \in I'}\left(v_i\right) > \frac{1}{k}, \\ &H2)\ LB\left(I\right) \geq L_1\left(I\right),\ \forall I, \\ &H3)\ LB(I^*) \leq LB(I), \forall I \end{aligned}$$

where L_1 is the bound given by (6) and I^* is the sub-instance of I defined as $I^* = \{i \in I : v_i > \frac{1}{k}\}$, the asymptotic worst-case of LB is

$$r_\infty\left(LB\right) \geq \frac{k}{k+1}.$$

We apply this general result to derive the asymptotic worst-case of the lower bounds previously introduced.

Theorem 5 *Any lower bound of the family LB_{TRLB} defined by (13) has an asymptotic worst-case at least equal to $\frac{3}{4}$.*

Proof. By Theorem 1, $TRLB$ solves to optimality all the instances with $v_i > \frac{1}{3}$. Moreover, the definition of the lower bounds ensures that the Hypothesis H3 of Theorem 4 is satisfied. Thus, one can assert that

$$r_\infty\left(LB_{TRLB}\right) \geq \frac{3}{4}.$$

■

Theorem 6 *The asymptotic worst-case of the lower bounds LB_{TRLB_1}, $LB^{(2)}_{TRLB_2}$, LB_{TRLB_3}, and $LB^{(2)}_{DFFR}$ is $\frac{3}{4}$.*

Proof. By Theorem 5, we have

$$\begin{aligned} r_\infty\left(LB_{TRLB_2}\right) &\geq \frac{3}{4} \\ r_\infty\left(LB^{(2)}_{TRLB_{FS}}\right) &\geq \frac{3}{4} \\ r_\infty\left(LB_{TRLB_{LLM}}\right) &\geq \frac{3}{4} \\ r_\infty\left(LB^{(2)}_{DFFR}\right) &\geq \frac{3}{4}. \end{aligned}$$

To prove that the equality holds for the four bounds, consider instances with $3k$ items of volume $v_i = \frac{1}{4} + \frac{1}{k}$. These instances need at least k bins. When the size and, consequently, k increase, B_{TMTRP} is not able to reduce the instance size, while $L_2 = L^{(2)}_{FS} = LB^{(2)}_{CPPT\_RED} = L_1 = \left\lceil \frac{3}{4}k + 3 \right\rceil$. ■

5 Computational Experiments

We presented in Section 4 a theoretical performance measure of the new $BPFB$s based on worst-case analysis. In this section, we analyze the computational performance of the new lower bounds on a large set of problem instances and compare these results to those of known $BPFB$s. More precisely, we compare the fast lower bounds $LB^{*(p)}_{DFFR}$, LB_{TRLB_2}, $LB^{(p)}_{TRLB_{FS}}$, and $LB_{TRLB_{LLM}}$ to the lower bounds L_1, L_2, $L^{*(p)}_{FS}$, and L_{LLM}. The results obtained by the bound L_3 are also presented in order to compare the Fast Lower Bounds with more complex ones. Optimal solutions have been obtained by using the Branch-and-Bound algorithm presented in [15].

The $BPFB$ $L_{CS}(m), m \leq 3$, derived from (12) is not included because $L_{CS}(1) = L_1$, $L_{CS}(2)$ is dominated by L_2, and $L_{CS}(3)$ is ominated by $LB^{*(p)}_{DFFR}$, LB_{TRLB_2}, $LB^{(p)}_{TRLB_{FS}}$, and $LB_{TRLB_{LLM}}$.

All bounds were implemented in C. Tests were conducted on a Pentium3 personal computer with 700Mhz clock and 256 Mb of Ram. Two types of benchmark problem sets have been used:

- Uniform distribution: instance sets based on a uniform distribution of the item volumes as in [15];
- Hard instances: hard-to-solve instances derived from specific sets collected in previous works by Scholl *et al.* [18], Schwerin and Wäscher [19], and Wäscher and Gau [21]. The ORLIB [2] instances are not considered, because the bound L_1 solves to optimality 159 instances out of 160, which is a clear indication that these instances are too easy and do not make a good benchmark. The TRIPLETS instances of Falkenauer [8] are discarded for the same reason: by construction, the value of the bound L_1 equals the optimal value.

5.1 Uniform distribution instances

The benchmark problem instances are generated according to the following parameters:

- Number of items: 50, 100, 300, 1000.
- Volume size class: $S_1 = [1, 100]$, $S_2 = [10, 100]$, $S_3 = [1, 90]$, $S_4 = [1, 80]$, $S_5 = [20, 100]$, $S_6 = [20, 80]$, $S_7 = [20, 70]$, $S_8 = [20, 60]$, $S_9 = [20, 50]$, $S_{10} = [20, 40]$, $S_{11} = [20, 35]$, and $S_{12} = [20, 30]$.
- Bin capacity: 100.

We use the volume classes defined by both Martello and Toth [15] and Fekete and Schepers [9]. Martello and Toth use $S'_1 = [1, 100]$, $S'_2 = [20, 100]$, and $S'_3 = [20, 80]$ with bin sizes $c = 100$, 120, and 150. After normalizing the volumes in the interval $\{1, 100\}$, these classes are a sub-set of the sets we use.

Relative to the tests performed by Fekete and Schepers, we add the set $[20, 100]$. While it is true that this set is covered by Theorem 1, the computational experiments show that it is interesting to consider it in order to compare how the bounds $L_{FS}^{*(20)}$ and $LB_{DFFR}^{(20)}$ perform near their optimality region.

We consider instances with 50, 100, 300, and 1000 items. 1000 random instances are generated for each combination of volume size and number of items. The seed for the random generation is changed every 100 instances.

5.2 Hard instances

We also consider the following instance classes known to be difficult to solve [1]:

- Test problems from Scholl *et al.* [18].
  - set_1: 720 instances with n = 50, 100, 200, and 500, bin capacity C = 100, 120, and 150, and weights uniformly generated from the intervals [1, 100], [20, 100], and [30, 100]. Those instances are built similarly to some of those proposed by Martello and Toth [16].
  - set_2: 480 instances with n = 50, 100, 200, and 500 and bin capacity C = 1000, generated to accommodate more items (three to nine items per bin in the average) in their optimal solutions than those in set 1.
  - set_3: ten difficult instances with n = 200, bin capacity C = 100000, and items weights in the range [20000; 35000].
- Test problems from Schwerin and Waescher [19].
  - was_1: 100 instances with bin capacity C = 1000, n = 100, minimum item weight equal to 150 and maximum item weight equal to 200.
  - was_2: 100 instances with bin capacity C = 1000, n = 120, minimum item weight equal to 150 and maximum item weight equal to 200.
- Test problems from Waescher and Gau [21]: 17 instances with the same parameters as for the was_1 and was_2 sets.

5.3 Analysis of results

The results obtained for the uniform distribution instances are reported in Tables 1, 2, 3, and 4, while Tables 5 and 6 report the results obtained for the hard instances.

The results of the bounds LB_{TRLB_2} and $LB_{TRLB_{LLM}}$ are generally equal to those of the lower bounds L_2 and L_{LLM} on all problem instances, and are therefore not included in the tables.

Tables 1, 2, 3 and 4 report the computational results on the uniform distribution instances. Tables 1 and 2 compare the mean relative gaps of the bounds L_1, L_2, L_{LLM}, L_3, $L_{FS}^{*(p)}$, $LB_{TRLB_{FS}}^{(p)}$, and $LB_{DFFR}^{*(p)}$ relative to the optimum. The volume size class and the number of items are reported in the first two columns, while the remaining columns display the relative gap of each bound. Following Fekete and Schepers, we present results for $p = 20$ and 100 for the bound $L_{FS}^{*(p)}$, while for the bounds $LB_{TRLB_{FS}}^{(p)}$ and $LB_{DFFR}^{*(p)}$, results are presented for p equal to 20 only. Tables 3 and 4 show, for each bound, the number of instances (out of 1000) for which the lower bound is equal to the optimum.

The results indicate that $L_{FS}^{*(p)}$ outperforms L_1, L_2, and L_{LLM}, while the new bounds $LB_{TRLB_{FS}}^{(p)}$ and $LB_{DFFR}^{*(p)}$ are able to improve the results of $L_{FS}^{*(p)}$ by both reducing the mean gap and increasing the number of instances solved to optimality. In particular, $LB_{DFFR}^{*(p)}$ outperforms $L_{FS}^{*(p)}$ even if we are executing 100 iterations for the latter and only 20 for $LB_{DFFR}^{*(p)}$, reducing the number of iterations by a factor of 5. The time required to compute the bounds is quite small for all of them (at most 10^{-2} seconds for the instances with 1000 items). Actually, the execution times for problem instances with a small number of items is negligible and, thus, only the mean execution times for the instances with 1000 items are included in Table 7.

The results for L_{LLM} present the same behavior as L_2 on all instances, except those with 50 items and volumes between 20% and 50% of the bin. In this case, the percentage deviation from the optimal solutions is reduced from 5.4735% to 5.4683%. This behaviour is related to the structure of the instances where L_{LLM} is able to improve L_2. Indeed, from the definitions of L_2 (equation (7)) and L_{LLM} (equation (8)), when the set I_3 of L_{LLM} is empty, L_{LLM} reduces to L_2. Otherwise, computing L_{LLM} for $\epsilon \in [0, \frac{1}{2}]$, such that $L_1\{I_2\{\epsilon\}\} \geq \lceil |I_3/2| \rceil$, we obtain

$$\left\lceil \frac{|I_3|}{2} \right\rceil + (|I(1-\epsilon,1)| + \max\{0, L_1(I_2(\epsilon)) - \lceil |I_3|/2 \rceil\}) =$$
$$|I(1-\epsilon,1)| + L_1(I_2(\epsilon)) = L_2(I,\epsilon),$$

while for $\epsilon \in [0, \frac{1}{2}]$ such that $L_1\{I_2\{\epsilon\}\} < \lceil |I_3/2| \rceil$, we obtain

$$\left\lceil \frac{|I_3|}{2} \right\rceil + (|I(1-\epsilon,1)| + \max\{0, L_1(I_2(\epsilon)) - \lceil |I_3|/2 \rceil\}) > L_2(I,\epsilon).$$

Because both bounds are computed as maxima over $\epsilon \in [0, 1/2]$, L_{LLM} improves L_2 if and only if the value of L_{LLM} is obtained for an $\epsilon = \epsilon_1$ such that $L_1\{I_2\{\epsilon_1\}\} < \lceil |I_3/2| \rceil$ and no other value ϵ_2 exists such that $L_2(I, \epsilon_2) = L_{LLM}$. Unfortunately, according to our tests, this situation rarely occurs for randomly built instances, in particular for instances with less than 100000 items. Indeed, Bourjolly and Rebetez obtained an improvement only for 15 very large instances with 200000 items [3].

According to [15], [4], and [9], the limits of the fast lower bounds for the BP problem become apparent for instances where the item volumes are neither small nor large. For those instances, however, the volume bound

L_1 usually finds good results, results that the other bounds are not able to improve upon. It is thus worth emphasizing that the bound $LB^{*(p)}_{DFFR}$ is able to significantly improve the results on the classes of volumes $\{20, 50\}$ and $\{20, 40\}$, while it obtains the same values as the other bounds on the classes $\{20, 35\}$ and $\{20, 30\}$.

Tables 5 and 6 display the computational results for the hard instances. Table 5 compares the mean relative optimality gaps of the bounds L_1, L_2, L_{LLM}, L_3, $L^{*(p)}_{FS}$, $LB^{(p)}_{TRLB_{FS}}$, and $LB^{*(p)}_{DFFR}$. The set name is reported in the first column, while the remaining columns display the relative gap of each bound. Following Fekete and Schepers, results are displayed for $p = 20$ and 100 for the bound $L^{*(p)}_{FS}$, while for the bounds $LB_{TRLB_{FS}}$ and $LB^{*(p)}_{DFFR}$, only results for p equal to 20 are presented.

Table 6 shows, for each bound, the number of instances for which the lower bound is equal to the optimum. The set name and the number of instances in each set are reported in the first two columns, while the remaining columns display the number of instances for which each bound is able to find the optimal number of bins.

The results show that all the $BPFB$s are outperformed by the more complex bound L_3 and obtain, except for L_1 and L_2, the same results. This situation marks the limits of the $BPFB$s. Indeed, when applied to known hard-to-solve instances, they pay their lower complexity by a lower accuracy. However, the bound $LB^{*(p)}_{DFFR}$ is able to improve the results of the other $BPFB$s on some hard instances, in particular sets 1 and 2 of the Scholl's instances. This is remarkable if one recalls that, usually, all $BPFB$s behave similarly to L_1 for those instances. The same behavior can be observed for the instances solved to optimality.

As stated before, from a computational point of view, the time needed to compute the different bounds is negligible (at most 10^{-2} seconds for 1000-item instances). One may argue, however, that $LB^{*(p)}_{DFFR}$ will require more time than $L^{*(p)}_{FS}$ when the size of the problem instances increases, because the procedure CPPT_RED has a complexity that can be dominated by the constant q^3 when q is high. From a practical point of view, however, the mean computational effort of the procedure CPPT_RED up to $q = 100$ is at most 3 times that of $LB^{*(p)}_{FS}$. Moreover, our results show that $LB^{*(p)}_{DFFR}$ achieves for $p = 20$ the same results that $LB^{*(p)}_{FS}$ obtains with $p = 100$. In this case, $LB^{*(20)}_{DFFR}$ requires about half the time needed by $LB^{*(100)}_{FS}$ to solve the same problem.

6 Conclusions

We introduced two new classes of fast lower bounds that combine and extend existing $BPFB$. The derived fast lower bounds have an asymptotic worst case of 3/4 independently on the $BPFB$ they extend. Extensive computational experiments have been run on a large data set, comparing existing and new $BPFB$.

These experiments indicate that one of the proposed $BPFB$, $LB^{*(p)}_{DFFR}$, is

able to obtain the best results among existing $BPFB$ relative to the deviation of the bound from the optimum. In particular, $LB^{*(p)}_{DFFR}$ is the only $BPFB$ able to improve the results in the Scholl *et al.* instances.

7 Acknowledgments

The authors wish to thank Professors Paolo Toth, Silvano Martello, and Alberto Caprara for their support and encouragement, as well as the anonymous referees for their helpful comments.

This research has been supported by ASI, the Italian Space Agency, under the ICARO Project, n. ASI I/R/137/01.

Partial funding for this project has also been provided by the Natural Sciences and Engineering Council of Canada, through its Discovery Research Grant program.

While working on this project, Dr. T.G. Crainic was Adjunct Professor at the Département d'informatique et de recheche opérationnelle of the Université de Montréal.

References

[1] A. C. F. Alvim, C. C. Ribeiro, F. Glover, D. J. Aloise, A hybrid improvement heuristic for the one-dimensional bin packing problem, Journal of Heuristics 10 (2) (2004) 205–229.

[2] J. E. Beasley, Or-library: distributing test problems by electronic mail, Journal of the Operational Research Society 41 (11) (1990) 1069–1072.

[3] J. M. Bourjolly, V. Rebetez, An analysis of lower bound procedures, Computers and Operations Research 32 (3) (2005) 395–405.

[4] H. Chao, M. P. Harper, R. W. Quong, A tight lower bound for optimal bin packing, Operations Research Letters 18 (3) (1995) 133–138.

[5] B. Chen, B. Srivastava, An improved lower bound for the bin packing problem, Discrete Applied Mathematics 66 (1996) 81–94.

[6] E. G. Coffman, M. R. Garey, D. S. Johnson, Bin packing approximation algorithms: a survey, in: D. S. Hochbaum (Ed.), Approximation Algorithms for NP-Hard Problems, PWS Publishing Company, Boston, MA, 1997 46–93.

[7] T. G. Crainic, G. Perboli, M. Pezzuto, R. Tadei, Computing the asymptotic worst-case of bin packing lower bounds, European Journal of Operational Research (forthcoming).

[8] E. Falkenauer, A hybrid grouping genetic algorithm for bin packing, Journal of Heuristics 2 (1) (1996) 5–30.

[9] S. P. Fekete, J. Schepers, New classes of lower bounds for bin packing problems, Mathematical Programming 91 (1) (2001) 11–31.

[10] M. R. Garey, D. S. Johnson, Computers and Intractability, W. H. Freeman and Co., New York, USA, 1979.

[11] P. C. Gilmore, R. E. Gomory, A linear programming approach to the cutting stock problem, Operations Research 9 (1961) 849–859.

[12] P. C. Gilmore, R. E. Gomory, A linear programming approach to the cutting stock problem - part ii, Operations Research 11 (1963) 863–888.

[13] M. Haouaria, A. Gharbia, Fast lifting procedures for the bin packing problem, Discrete Optimization 2 (2005) 201–218.

[14] M. Labbé, G. Laporte, H. Mercure, Capacitated vehicle routing on trees, Operations Research 39 (1991) 616–622.

[15] S. Martello, P. Toth, Knapsack Problems - Algorithms and computer implementations, John Wiley & Sons, Chichester, UK, 1990.

[16] S. Martello, P. Toth, Lower bounds and reduction procedures for the bin packing problem, Discrete Applied Mathematics 28 (1) (1990) 59–70.

[17] G. Perboli, Bounds and heuristics for the packing problems, Ph.D. thesis, Politecnico di Torino, available at http://www.orgroup.polito.it/People/perboli/phd-thesys.pdf (2002).

[18] A. Scholl, R. Klein, C. Jürgens, Bison: A fast hybrid procedure for exactly solving the one-dimensional bin packing problem, Computers and Operations Research 24 (7) (1997) 627–645.

[19] P. Schwerin, G. Wäscher, The bin-packing problem: A problem generator and some numerical experiments with ffd packing and mtp, International Transactions in Operational Research 4 (5/6) (1997) 377–389.

[20] F. Vanderbeck, Computational study of a column generation algorithm for bin packing and cutting stock problems, Mathematical Programming 86 (3) (1999) 565–594.

[21] G. Wäscher, T. Gau, Heuristics for the integer one-dimensional cutting stock problem: A computational study, OR Spektrum 18 (1996) 131–144.

Set	n	L_1	L_2	L_{LLM}	L_3	$L_{FS}^{*(20)}$	$L_{FS}^{*(100)}$	$LB_{TRLB_{FS}}^{(20)}$	$LB_{DFFR}^{*(20)}$
{1,100}	50	5,0665	0,6554	0,6554	0,2147	0,3886	0,3886	0,3886	0,3802
	100	3,9539	0,5499	0,5499	0,1951	0,3351	0,3351	0,3329	0,3251
	300	2,5056	0,3647	0,3647	0,1225	0,2067	0,2053	0,2067	0,1954
	500	1,9992	0,2659	0,2659	0,0879	0,155	0,1526	0,155	0,1418
	1000	1,4437	0,207	0,207	0,0698	0,1184	0,1166	0,1184	0,1105
{1,90}	50	4,8326	0,6961	0,6961	0,2476	0,3976	0,3976	0,3976	0,3811
	100	3,3923	0,5402	0,5402	0,1962	0,3152	0,3152	0,3152	0,2917
	300	1,7277	0,2847	0,2847	0,0911	0,1537	0,1523	0,1537	0,1428
	500	1,2107	0,1922	0,1922	0,0653	0,105	0,105	0,105	0,0985
	1000	0,6332	0,099	0,099	0,0324	0,0426	0,0426	0,0426	0,0411
{1,80}	50	2,8282	0,692	0,692	0,4771	0,5601	0,5601	0,5601	0,5553
	100	1,3768	0,411	0,411	0,3119	0,3254	0,3254	0,3254	0,3231
	300	0,3366	0,1427	0,1427	0,1318	0,1333	0,1333	0,1333	0,1333
	500	0,1419	0,0877	0,0877	0,0853	0,0853	0,0853	0,0853	0,0853
	1000	0,0377	0,0372	0,0372	0,0372	0,0372	0,0372	0,0372	0,0372
{10,100}	50	5,9236	0,6042	0,6042	0,1557	0,3211	0,3211	0,3211	0,3177
	100	4,7013	0,4851	0,4851	0,1854	0,302	0,3002	0,302	0,2948
	300	3,2794	0,3459	0,3459	0,1239	0,197	0,1958	0,197	0,1831
	500	2,8287	0,2795	0,2795	0,0959	0,1568	0,155	0,1568	0,1485
	1000	2,2811	0,2148	0,2148	0,0773	0,1208	0,1201	0,1208	0,114
{20,100}	50	8,2331	0,5786	0,5786	0,1494	0,304	0,304	0,304	0,2974
	100	7,252	0,4964	0,4964	0,0997	0,2547	0,2531	0,2547	0,2338
	300	5,9705	0,3256	0,3256	0,0808	0,167	0,1648	0,167	0,1557
	500	5,5259	0,2559	0,2559	0,0614	0,1333	0,1327	0,1333	0,1226
	1000	5,0315	0,2011	0,2011	0,049	0,1017	0,1012	0,1017	0,0925
{20,80}	50	6,0511	0,9357	0,9357	0,2127	0,4346	0,4305	0,4346	0,39
	100	4,5131	0,7401	0,7401	0,2095	0,3796	0,3775	0,3796	0,3317
	300	2,5733	0,4818	0,4818	0,1501	0,2641	0,2634	0,2641	0,2393
	500	2,0135	0,4018	0,4018	0,132	0,2119	0,2107	0,2119	0,1906
	1000	1,3241	0,3105	0,3105	0,1047	0,1647	0,1641	0,1647	0,1504
{20,70}	50	3,3755	2,3641	2,3641	1,4705	1,7752	1,7752	1,7706	1,6028
	100	2,3986	2,1616	2,1616	1,5577	1,7237	1,7237	1,7214	1,6212
	300	1,822	1,8192	1,8192	1,5134	1,5735	1,5735	1,5735	1,5607
	500	1,7275	1,7275	1,7275	1,4842	1,5344	1,5344	1,5344	1,5331
	1000	1,633	1,633	1,633	1,4681	1,5365	1,5365	1,5365	1,5363

Table 1: Uniform distribution instances - Mean relative optimality gaps (1000 instances)- Part 1

Set	n	L_1	L_2	L_{LLM}	L_3	$L_{FS}^{*(20)}$	$L_{FS}^{*(100)}$	$LB_{TRLB_{FS}}^{(20)}$	$LB_{DFFR}^{*(20)}$
{20,60}	50	2,9006	2,9006	2,9006	2,7739	2,8559	2,8559	2,8511	2,6855
	100	2,6206	2,6206	2,6206	2,6089	2,6139	2,6139	2,6139	2,5914
	300	2,0841	2,0841	2,0841	2,0841	2,0841	2,0841	2,0841	2,0833
	500	1,9839	1,9839	1,9839	1,9839	1,9839	1,9839	1,9839	1,9839
	1000	1,9066	1,9066	1,9066	1,9066	1,9066	1,9066	1,9066	1,9066
{20,50}	50	5,4735	5,4735	5,4683	5,464	5,4485	5,4485	5,4485	4,8155
	100	5,982	5,982	5,982	5,982	5,982	5,982	5,982	5,3776
	300	6,3165	6,3165	6,3165	6,3165	6,3165	6,3165	6,3165	5.8552
	500	6,3685	6,3685	6,3685	6,3685	6,3685	6,3685	6,3685	5,8916
	1000	6,3799	6,3799	6,3799	6,3799	6,3799	6,3799	6,3799	5,9317
{20,40}	50	3,6376	3,6376	3,6376	3,6376	3,6376	3,6376	3,6376	3,2607
	100	3,8339	3,8339	3,8339	3,8339	3,8339	3,8339	3,8339	3,5721
	300	4,0625	4,0625	4,0625	4,0625	4,0625	4,0625	4,0625	4,0038
	500	4,1036	4,1036	4,1036	4,1036	4,1036	4,1036	4,1036	4,0817
	1000	4,1446	4,1446	4,1446	4,1446	4,1446	4,1446	4,1446	4,1388
{20,35}	50	6,4165	6,4165	6,4165	6,4165	6,4165	6,4165	6,4165	6,4165
	100	6,8002	6,8002	6,8002	6,8002	6,8002	6,8002	6,8002	6,8002
	300	7,1046	7,1046	7,1046	7,1046	7,1046	7,1046	7,1046	7,1046
	500	7,1424	7,1424	7,1424	7,1424	7,1424	7,1424	7,1424	7,1424
	1000	7,2003	7,2003	7,2003	7,2003	7,2003	7,2003	7,2003	7,2003
{20,30}	50	1,5041	1,5041	1,5041	1,5041	0,7349	0,7349	0,7349	0,7349
	100	0,4903	0,4903	0,4903	0,4903	0,3543	0,3543	0,3543	0,3543
	300	1,3353	1,3353	1,3353	1,3353	1,1366	1,1366	1,1366	1,1366
	500	1,5888	1,5888	1,5888	1,5888	1,4344	1,4344	1,4344	1,4344
	1000	1,5962	1,5962	1,5962	1,5962	1,4873	1,4873	1,4873	1,4873

Table 2: Uniform distribution instances - Mean relative optimality gaps (1000 instances) - Part 2

Set	n	L_1	L_2	L_{LLM}	L_3	$L_{FS}^{*(20)}$	$L_{FS}^{*(100)}$	$LB_{TRLB_2}^{(20)}$	$LB_{DFFR}^{*(20)}$
{1,100}	50	172	830	830	947	901	901	901	903
	100	82	719	719	903	830	830	831	835
	300	19	510	510	824	712	713	712	725
	500	17	460	460	792	658	660	658	680
	1000	2	389	389	734	606	607	606	618
{1,90}	50	287	835	835	945	908	908	908	912
	100	214	751	751	914	858	858	858	869
	300	244	645	645	878	795	797	795	809
	500	280	643	643	864	787	787	787	798
	1000	398	672	672	863	830	830	830	834
{1,80}	50	550	853	853	902	883	883	883	884
	100	627	830	830	874	868	868	868	869
	300	738	826	826	840	838	838	838	838
	500	786	823	823	828	828	828	828	828
	1000	849	850	850	850	850	850	850	850
{10,100}	50	67	830	830	958	911	911	911	912
	100	8	731	731	899	833	834	833	837
	300	0	493	493	803	689	691	689	710
	500	0	420	420	759	644	646	644	656
	1000	0	343	343	703	585	585	585	599
{20,100}	50	1	816	816	954	904	904	904	906
	100	0	689	689	939	841	842	841	854
	300	0	504	504	859	714	717	714	733
	500	0	431	431	829	673	673	673	691
	1000	0	343	343	772	610	611	610	630
{20,80}	50	134	761	761	948	891	892	891	902
	100	72	645	645	899	811	812	811	835
	300	5	455	455	811	672	673	672	701
	500	0	387	387	740	627	627	627	648
	1000	0	325	325	675	552	552	552	573
{20,70}	50	301	453	453	660	588	588	589	628
	100	85	137	137	319	280	280	281	308
	300	1	2	2	10	35	35	35	35
	500	0	0	0	0	13	13	13	13
	1000	0	0	0	0	0	0	0	0

Table 3: Uniform distribution instances - Instances with zero gap (out of 1000 instances) - Part 1

Set	n	L_1	L_2	L_{LLM}	L_3	$L_{FS}^{*(20)}$	$L_{FS}^{*(100)}$	$LB_{TRLB_2}^{(20)}$	$LB_{DFFR}^{*(20)}$
{20,60}	50	400	400	400	422	409	409	410	421
	100	87	87	87	88	87	87	87	93
	300	0	0	0	0	0	0	0	0
	500	0	0	0	0	0	0	0	0
	1000	0	0	0	0	0	0	0	0
	1000	0	0	0	0	0	0	0	0
{20,50}	50	131	131	131	131	131	131	131	174
	100	2	2	2	2	2	2	2	13
	300	0	0	0	0	0	0	0	0
	500	0	0	0	0	0	0	0	0
	1000	0	0	0	0	0	0	0	0
{20,40}	50	418	418	418	418	418	418	418	439
	100	28	28	28	28	28	28	28	76
	300	0	0	0	0	0	0	0	4
	500	0	0	0	0	0	0	0	1
	1000	0	0	0	0	0	0	0	0
{20,35}	50	63	63	63	63	63	63	63	63
	100	0	0	0	0	0	0	0	0
	300	0	0	0	0	0	0	0	0
	500	0	0	0	0	0	0	0	0
	1000	0	0	0	0	0	0	0	0
{20,30}	50	809	809	809	909	909	909	909	909
	100	890	890	890	924	924	924	924	924
	300	163	163	163	172	172	172	172	172
	500	7	7	7	7	7	7	7	7
	1000	0	0	0	0	0	0	0	0

Table 4: Uniform distribution instances - Instances with zero gap (out of 1000 instances) - Part 2

Set	L_1	L_2	L_{LLM}	L_3	$L_{FS}^{*(20)}$	$L_{FS}^{*(100)}$	$LB_{TRLB_2}^{(20)}$	$LB_{DFFR}^{*(20)}$
Scholl 1	5,0791	1,5163	1,5163	0,5651	1,3239	1,3235	1,3239	1,3078
Scholl 2	4,8495	4,8495	4,8495	2,0381	4,8245	4,8245	4,8245	4,7191
Scholl 3	8,5593	8,5593	8,5593	6,8814	8,5593	8,5593	8,5593	8,5593
Schwerin and Waescher 1	6,8977	6,8977	6,8977	4,7368	6,8977	6,8977	6,8977	6,8977
Schwerin and Waescher 2	8,4881	8,4881	8,4881	6,0889	8,4881	8,4881	8,4881	8,4881
Waescher and Gau	6,573	6,573	6,573	5,8727	6,573	6,573	6,573	6,573

Table 5: Hard instances - Mean relative optimality gaps

Set	Inst.	L_1	L_2	L_{LLM}	L_3	$L_{FS}^{*(20)}$	$L_{FS}^{*(100)}$	$LB_{TRLB_2}^{(20)}$	$LB_{DFFR}^{*(20)}$
Scholl 1	720	65	244	244	525	296	297	296	298
Scholl 2	480	92	92	92	280	92	92	92	98
Scholl 3	10	0	0	0	0	0	0	0	0
Schwerin and Waescher 1	100	9	9	9	10	9	9	9	9
Schwerin and Waescher 2	100	0	0	0	4	0	0	0	0
Waescher and Gau	17	0	0	0	0	0	0	0	0

Table 6: Hard instances - Instances with zero gap

Set	L_1	L_2	L_{LLM}	$L_{FS}^{*(20)}$	$L_{FS}^{*(100)}$	$LB_{TRLB_2}^{(20)}$	$LB_{DFFR}^{*(20)}$
{1,100}	0.0000	0.0001	0.0001	0.0021	0.0109	0.0032	0.0052
{1,90}	0.0000	0.0002	0.0002	0.0051	0.0221	0.0046	0.0132
{1,80}	0.0000	0.0001	0.0001	0.0022	0.0099	0.0031	0.0053
{10,100}	0.0000	0.0001	0.0002	0.0019	0.0113	0.0032	0.0055
{20,100}	0.0000	0.0001	0.0001	0.0024	0.0115	0.0027	0.0048
{20,80}	0.0000	0.0001	0.0002	0.0020	0.0108	0.0028	0.0055
{20,70}	0.0000	0.0002	0.0002	0.0055	0.0245	0.0062	0.0166
{20,60}	0.0000	0.0001	0.0001	0.0022	0.0113	0.0032	0.0062
{20,50}	0.0000	0.0001	0.0001	0.0020	0.0119	0.0031	0.0063
{20,40}	0.0000	0.0001	0.0001	0.0021	0.0110	0.0030	0.0052
{20,35}	0.0000	0.0002	0.0002	0.0045	0.0212	0.0047	0.0100
{20,30}	0.0000	0.0002	0.0002	0.0041	0.0224	0.0055	0.0087

Table 7: Uniform distribution instances - Mean execution times for the instances with 1000 items

Annex: Procedures

Algorithm 1 TMTRP

Input I : Instance of the BP problem ordered by non-increasing values of v_j

$head$: beginning of the list;
$tail$: end of the list;
j : last j found;
$usedbins$: minimum number of bins to use

$head = 1$
$tail = N$
$j = k = N$
$usedbins = 0$
For the dominance criterion D1
while $(head < tail)$ **and** $(1 - v_{head} < v_{tail})$ **do**
  $usedbins = usedbins + 1$
  $I \setminus \{head\}$
  $head = head + 1$
end while
For the dominance criterion D2
while $(head < tail)$ **and** $(head < j)$ **and** $(1 - v_{head} - v_{tail} < v_{tail-1})$ **do**
  $usedbins = usedbins + 1$
  while $(v_{head} + v_{j-1} \leq 1)$ **and** $(head < j)$ **do**
    $k = j$
    $j = j + 1$
  end while
  $head = head + 1$
  if $k = tail$ **then**
    $tail = tail - 1$
  end if
  $I \setminus \{head, k\}$
  $j = j - 1$
end while
return I

Algorithm 2 CPPT_RED

Input I : Instance of the BPP ordered by non-increasing values of v_i
Input k : parameter for the dual feasible function $u^{(k)}$

$usedbins$: number of used bins
lb : minimum number of bins to use;
$lclass$: list of loading classes in the instance;

$lb = 0$
$usedbins = 0$
We apply the $u^{(k)}(x)$ dual feasible function
for all $i \in I$ **do**
  if $\nexists lclass[v_i]$ **then**
    add the class $u^{(k)}(v_i)$ to $lclass$
  end if
  $lclass[v_i] = lclass[v_i] + 1$
end for
$cont =$ **true**
while $lclass \neq \emptyset$ **do**
  $I' = I$
  $lclass' = lclass$
  while not $cont$ **and** $I \neq \emptyset$ **do**
    $cont = false$
    if not $cont$ **then**
      $cont = FINDF1(I', lclass', usedbins)$
    end if
    if not $cont$ **then**
      $cont = FINDF2(I', lclass', usedbins)$
    end if
    if not $cont$ **then**
      $cont = FINDF3(I', lclass', usedbins)$
    end if
  end while
  $tvol = 0$
  for all $j \in lclass'$ **do**
    $tvol = j * lclass'[j]$
  end for
  $lb = \max\{lb, usedbins + \lceil tvol \rceil\}$
  remove the class $lclass[j]$ with the minimum j
end while
return lb

Algorithm 3 FINDF1

Input I : Instance of the BPP ordered by non-increasing values of v_i
Input $lclass$: list of loading classes in the instance
Input $usedbins$: number of used bins

$head = u^{(k)}(v_1)$
$tail = u^{(k)}(v_N)$
if $1 - head < tail1$ **then**
  $usedbins = usedbins + |lclass[u(head))]|$
  remove the class $head$ from $lclass$
  return true
else
  return false
end if

Algorithm 4 FINDF2

Input I : Instance of the BPP ordered by non-increasing values of v_i
Input $lclass$: list of loading classes in the instance
Input $usedbins$: number of used bins

if $|I| < 3$ **then**
  return false
end if
$head = u^{(k)}(v_1)$
$tail1 = u^{(k)}(v_N)$
$tail2 = u^{(k)}(v_{N-1})$
if $(1 - head - tail1 \geq 0)$ **and** $(1 - head - tail1 < tail2)$ **then**
  $j = \arg\min_{j \in lclass:\{1,j\} \text{ is a FS}} \{1 - (head + j)\}$
  if $j = head$ **then**
    $usedbins = usedbins + \lfloor |lclass[head]|/2 \rfloor$
    if $|lclass[head]| - 2 * \lfloor |lclass[head]|/2 \rfloor = 0$ **then**
      remove the class $head$ from $lclass$
    else
      $|lclass[head]| = 1$
    end if
  else
    $usedbins = usedbins + \min\{lclass[head], lclass[j]\}$
    $|lclass[head]| = |lclass[head]| - \min\{lclass[head], lclass[j]\}$
    $|lclass[j]| = |lclass[j]| - \min\{lclass[head], lclass[j]\}$
    Remove the classes between $lclass(head)$ and $lclass(j)$ with cardinality equal to 0
  end if
  return true
else
  return false
end if

Algorithm 5 FINDF3

Input I : Instance of the BPP ordered by non-increasing values of v_i
Input $lclass$: list of loading classes in the instance
Input $usedbins$: number of used bins

if $|I| < 4$ **then**
  return false
end if
$head = u^{(k)}(v_1)$; $tail1 = u^{(k)}(v_N)$; $tail2 = u^{(k)}(v_{N-1})$;
$tail3 = u^{(k)}(v_{N-2})$; $cont =$ **false**;
if $(1 - head - tail1 - tail2 \geq 0)$ **and** $(1 - head - tail1 - tail2 < tail3)$ **then**
  $j_1, j_2 = \arg\min_{j_1, j_2 \in lclass:\{1, j_1, j_2\} \text{ is a FS}} \{1 - (head + j_1 + j_2)\}$
  $j_3 = \arg\min_{j_3 \in lclass:\{1, j_3\} \text{ is a FS}} \{1 - (head + j_3)\}$
  if $j_3 \geq j_1 + j_2$ **then**
    The pair $(head, j_3)$ dominates the other bins
    if $j = head$ **then**
      $usedbins = usedbins + \lfloor |lclass[head]| / 2 \rfloor$
      if $|lclass[head]| - 2 * \lfloor |lclass[head]| / 2 \rfloor = 0$ **then**
        remove the class $head$ from $lclass$
      else
        $|lclass[head]| = 1$
      end if
    else
      $usedbins = usedbins + \min\{lclass[head], lclass[j_3]\}$
      $|lclass[head]| = |lclass[head]| - \min\{lclass[head], lclass[j_3]\}$
      $|lclass[j]| = |lclass[j]| - \min\{lclass[head], lclass[j_3]\}$
    end if
    $cont =$ **true**
  else
    if $(j_1 = j_3)$**or**$(j_2 = j_3)$ and case 2 or 3 of the dominance criterion D3 hold **then**
      The triplet $(head, j_1, j_2)$ dominates the other bins
      FINDF3_TRIPLET()
    end if
  end if
  if $cont =$ **true then**
    FINDF3_UPDATELIST()
  end if
end if
Remove the classes in $lclass$ with cardinality equal to 0
return cont

Algorithm 6 FINDF3_TRIPLET

$divHead = 1; divIT1 = 1; divIT2 = 1;$
if $j_1 = head$ **then**
  $divHead = divHead + 1; divIT1 = divIT1 - 1;$
end if
if $j_2 = head$ **then**
  $divHead = divHead + 1; divIT2 = divIT1 - 1;$
end if
if $(j_1 = j_2)$**and**$(j_1 \neq head)$ **then**
  $divIT1 = divIT1 + 1; divIT2 = divIT2 - 1;$
end if
$cont =$ **true**

Algorithm 7 FINDF3_UPDATELIST

$bin2add = \lfloor lclass[head)]/divHead \rfloor$
if $divIT1! = 0$ **then**
  $bin2add = \min\left(bin2add, \lfloor lclass\left(j_1\right)/divIT1 \rfloor\right)$
end if
if $divIT2! = 0$ **then**
  $bin2add = \min\left(bin2add, \lfloor lclass\left(j_2\right)/divIT2 \rfloor\right)$
end if
$usedbins = usedbins + bin2add$
$lclass\left(head\right) = lclass\left(head\right) - divHead * bin2add$
$lclass\left(j_1\right) = lclass\left(j_1\right) - divIT1 * bin2add$
$lclass\left(j_1\right) = lclass\left(j_2\right) - divIT2 * bin2add$