New Classes of Lower Bounds for Bin Packing Problems [⋆]

Sándor P. Fekete and Jörg Schepers

Center for Parallel Computing, Universität zu Köln
D–50923 Köln, Germany
{fekete, schepers}@@zpr.uni-koeln.de

Abstract. The bin packing problem is one of the classical NP-hard optimization problems. Even though there are many excellent theoretical results, including polynomial approximation schemes, there is still a lack of methods that are able to solve practical instances optimally. In this paper, we present a fast and simple generic approach for obtaining new lower bounds, based on dual feasible functions. Worst case analysis as well as computational results show that one of our classes clearly outperforms the currently best known "economical" lower bound for the bin packing problem by Martello and Toth, which can be understood as a special case. This indicates the usefulness of our results in a branch and bound framework.

1 Introduction

The bin packing problem (BPP) can be described as follows: Given a set of n "items" with integer size $x_1, \ldots, x_n$, and a supply of identical "containers" of capacity C, decide how many containers are necessary to pack all the items. This task is one of the classical problems of combinatorial optimization and NP-hard in the strong sense – see Garey and Johnson [9]. An excellent survey by Coffmann, Garey, and Johnson can be found as Chapter 2 in the recent book [2].

Over the years, many clever methods have been devised to deal with the resulting theoretical difficulties. Most notably, Fernandez de la Vega and Lueker [8], and Karmarkar and Karp [12] have developed polynomial time approximation schemes that allow it to approximate an optimal solution within $1+\varepsilon$ in polynomial (even linear) time, for any fixed ε. However, these methods can be hardly called practical, due to the enormous size of the constants. On the other hand, there is still a lack of results that allow it solve even moderately sized test problems optimally – see Martello and Toth [15,16], and the ORLIB set of benchmark problems [1]. The need for better understanding is highlighted by a recent observation by Gent [10]. He showed that even though some of these benchmark

[⋆] This work was supported by the German Federal Ministry of Education, Science, Research and Technology (BMBF, Förderkennzeichen 01 IR 411 C7).

problems of 120 and 250 items had defied all systematic algorithmic solution attempts, they are not exactly hard, since they can be solved by hand.

This situation emphasizes the need for ideas that are oriented towards the exact solution of problem instances, rather than results that are mainly worst case oriented. In this paper, we present a simple and fast generic approach for obtaining lower bounds, based on *dual feasible functions*. Assuming that the items are sorted by size, our bounds can be computed in linear time with small constants, a property they share with the best lower bound for the BPP by Martello and Toth [15,16]. As it turns out, one of our classes of lower bounds can be interpreted as a systematic generalization of the bound by Martello and Toth. Worst case analysis as well as computational results indicate that our generalization provides a clear improvement in performance, indicating their usefulness in the context of a branch and bound framework. Moreover, we can show that the simplicity of our systematic approach is suited to simplify and improve other hand-tailored types of bounds.

The rest of this paper is organized as follows. In Section 2, we give an introduction to dual feasible functions and show how they can be used to obtain fast and simple lower bounds for the BPP. In Section 3, we consider the worst case performance of one of our classes of lower bounds, as well as some computational results on the practical performance. Section 4 concludes with a discussion of possible extensions, including higher-dimensional packing problems.

2 Dual Feasible Functions

For the rest of this paper, we assume without loss of generality that the items have size $x_i \in [0,1]$, and the container size C is normalized to 1. Then we introduce the following:

Definition 1 (Dual feasible functions). *A function $u : [0,1] \to [0,1]$ is called* dual feasible*, if for any finite set S of nonnegative real numbers, we have the relation*

$$\sum_{x \in S} x \leq 1 \Longrightarrow \sum_{x \in S} u(x) \leq 1. \tag{1}$$

Dual feasible functions have been used in the performance analysis of heuristics for the BPP, first by Johnson [11], then by Lueker [13]; see Coffman and Lueker [3] for a more detailed description. The Term (which was first introduced by Lueker [13]) refers to the fact that for any dual feasible function u and for any bin packing instance with item sizes $x_1, \ldots, x_n$, the vector $(u(x_1), \ldots, u(x_n))$ is a feasible solution for the dual of the corresponding fractional bin packing problem (see [12]). By definition, convex combination and compositions of dual feasible functions are dual feasible.

We show in this paper that dual feasible functions can be used for improving lower bounds for the one-dimensional bin packing problem. This is based on the following easy lemma.

Lemma 1. *Let $I := (x_1, \ldots, x_n)$ be a BPP instance and let u be a dual feasible function. Then any lower bound for the transformed BPP instance $u(I) := (u(x_1), \ldots, u(x_n))$ is also a lower bound for I.*

By using a set of dual feasible functions $\mathcal{U}$ and considering the maximum value over the transformed instances $u(I)$, $u \in \mathcal{U}$, we can try to obtain even better lower bounds.

In [13], a particular class of dual feasible functions is described; it relies on a special rounding technique. For a given $k \in \mathbb{N}$, consider the stair function $u^{(k)}$ that maps (for $i \in \{1, \ldots, k\}$) all values from the interval $[\frac{i}{k+1}, \frac{i+1}{k+1})$ onto the value $\frac{i}{k}$. 1 is mapped to 1. We give a slightly improved version and a simple proof that these functions are dual feasible.

Theorem 1. *Let $k \in \mathbb{N}$. Then*

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

is a dual feasible function.

Proof. Let S be a finite set of nonnegative real numbers with $\sum_{x_i \in S} x_i \leq 1$. We have to show that $\sum_{x_i \in S} u^{(k)}(x_i) \leq 1$. Let $T := \{x_i \in S \mid x_i(k+1) \in \mathbb{Z}\}$. Clearly, we only need to consider the case $S \neq T$. Then

$$\begin{aligned}(k+1) \sum_{x_i \in T} u^{(k)}(x_i) + k \sum_{x_i \in S \setminus T} u^{(k)}(x_i) &= (k+1) \sum_{x_i \in T} x_i + \sum_{x_i \in S \setminus T} \lfloor (k+1)x_i \rfloor \\ &< (k+1) \sum_{x_i \in S} x_i .\end{aligned}$$

By definition of $u^{(k)}$, the terms $(k+1)\sum_{x \in T} u^{(k)}(x)$ and $k \sum_{x \in S \setminus T} u^{(k)}(x)$ are integer, so by virtue of $\sum_{x_i \in S} x_i \leq 1$, we have the inequality $(k+1)\sum_{x \in T} u^{(k)}(x) + k \sum_{x \in S \setminus T} u^{(k)}(x) \leq k$, implying $\sum_{x \in S} u^{(k)}(x) \leq 1$. □

The rounding mechanism is visualized in Figure 1, showing the difference of the stair functions $u^{(k)}$, $k \in \{1, \ldots, 4\}$, and the identity *id* over the interval $[0,1]$. For the BPP, we mostly try to increase the sizes by a dual feasible function, since this allows us to obtain a tighter bound. The hope is to find a $u^{(k)}$ for which as many items as possible are in the "win zones" – the subintervals of $[0,1]$ for which the difference is positive.

The following class of dual feasible functions is the implicit basis for the bin packing bound L_2 by Martello and Toth [15,16]. This bound is obtained by neglecting all items smaller than a given value ϵ. We account for these savings by increasing all items of size larger than $1-\epsilon$. Figure 2 shows the corresponding win and loss zones.

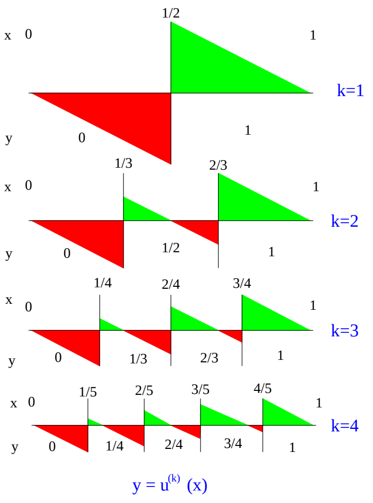

Fig. 1. Win and loss zones for $u^{(k)}$.

Theorem 2. *Let* $\epsilon \in [0, \frac{1}{2}]$. *Then*

$$U^{(\epsilon)} : [0,1] \longrightarrow [0,1]$$
$$x \mapsto \begin{cases} 1, & \textit{for } x > 1-\epsilon \\ x, & \textit{for } \epsilon \le x \le 1-\epsilon \\ 0, & \textit{for } x < \epsilon \end{cases}$$

is a dual feasible function.

Proof. Let S be a finite set of nonnegative real numbers, with $\sum_{x \in S} x \le 1$. We consider two cases. If S contains an element larger than $1-\epsilon$, then all other elements have to be smaller than ϵ. Hence we have $\sum_{x \in S} U^{(\epsilon)}(x) = 1$. If all elements of S have at most size $1-\epsilon$, we have $\sum_{x \in S} U^{(\epsilon)}(x) \le \sum_{x \in S} x \le 1$. □

Our third class of dual feasible functions has some similarities to some bounds that were hand-tailored for the two-dimensional and three-dimensional BPP by

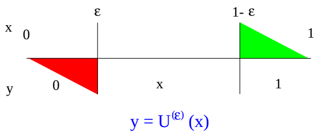

Fig. 2. Win and loss zones for $U^{(\epsilon)}$.

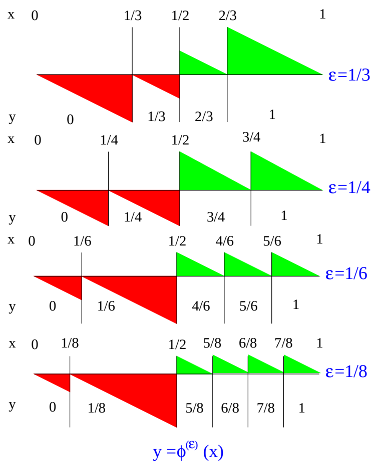

Fig. 3. Win and loss zones for $\phi^{(\epsilon)}$.

Martello and Vigo [17], and Martello, Pisinger, and Vigo [14]. However, our bounds are simpler and dominate theirs.

This third class also ignores items of size below a threshold value ϵ. For the interval $(\epsilon, \frac{1}{2}]$, these functions are constant, on $(\frac{1}{2}, 1]$ they have the form of stair functions. Figure 3 shows that for small values of ϵ, the area of loss zones for $\phi^{(\epsilon)}$ exceeds the area of win zones by a clear margin. This contrasts to the behavior of the functions $u^{(k)}$ and $U^{(\epsilon)}$, where the win and loss areas have the same size.

Theorem 3. *Let $\epsilon \in [0, \frac{1}{2})$. Then*

$$\phi^{(\epsilon)} : [0,1] \to [0,1]$$
$$x \mapsto \begin{cases} 1 - \frac{\lfloor (1-x)\epsilon^{-1} \rfloor}{\lfloor \epsilon^{-1} \rfloor}, & \textit{for } x > \frac{1}{2} \\ \frac{1}{\lfloor \epsilon^{-1} \rfloor}, & \textit{f"ur } \epsilon \le x \le \frac{1}{2} \\ 0, & \textit{for } x < \epsilon \end{cases}$$

is a dual feasible function.

Proof. Let S be a finite set of nonnegative real numbers, with $\sum_{x \in S} x \le 1$. Let $S' := \{x \in S \,|\, \epsilon \le x \le \frac{1}{2}\}$. We distinguish two cases:

If all elements of S have size at most $\frac{1}{2}$, then by definition of S',

$$1 \ge \sum_{x \in S} x \ge \sum_{x \in S'} x \ge |S'|\epsilon \tag{2}$$

holds. Since $|S'|$ is integral, it follows that $|S'| \leq \lfloor \epsilon^{-1} \rfloor$, hence

$$\sum_{x \in S} \phi^{(\epsilon)}(x) = \sum_{x \in S'} \phi^{(\epsilon)}(x) = |S'| \frac{1}{\lfloor \epsilon^{-1} \rfloor} \leq 1. \tag{3}$$

Otherwise S contains exactly one element $y > \frac{1}{2}$ and we have

$$1 \geq \sum_{x \in S} x \geq y + \sum_{x \in S'} x \geq y + |S'| \epsilon. \tag{4}$$

Therefore $|S'| \leq \lfloor (1-y)\epsilon^{-1} \rfloor$ and hence

$$\sum_{x \in S} \phi^{(\epsilon)}(x) = \phi^{(\epsilon)}(y) + \sum_{x \in S'} \phi^{(\epsilon)}(x) = 1 - \frac{\lfloor (1-y)\epsilon^{-1} \rfloor}{\lfloor \epsilon^{-1} \rfloor} + |S'| \frac{1}{\lfloor \epsilon^{-1} \rfloor} \leq 1. \tag{5}$$

□

3 A Class of Lower Bounds for the BPP

In this section, we show how dual feasible functions can be combined by virtue of Lemma 1 in order to get good bounds for the BPP. For this purpose, we consider the lower bound L_2, suggested by Martello an Toth [15,16], which can be computed in time $O(n \log n)$. According to the numerical experiments by Martello and Toth, L_2 yields very good approximations of the optimal value. We describe L_2 with the help of dual feasible functions. Using Lemma 1, it is straightforward to see that L_2 provides a lower bound for the BPP. Our description allows an easy generalization of this bound, which can be computed in linear time. We show that this generalized bound improves the asymptotic worst-case performance from $\frac{2}{3}$ to $\frac{3}{4}$. Empirical studies provide evidence that also the practical performance is significantly improved.

3.1 The Class $L_*^{(q)}$

We give a definition of *(asymptotic) worst case performance*:

Definition 2 ((Asymptotic) worst case performance). *Let L be a lower bound for a minimization problem P. Let $opt(I)$ denote the optimal value of P for an Instance I. Then*

$$r(L) := \sup \left\{ \frac{L(I)}{opt(I)} | I \textit{ is an instance of } P \right\} \tag{6}$$

is called the worst case performance *and*

$$r_\infty(L) := \inf_{s \in \mathbb{R}_0^+} \sup \left\{ \frac{L(I)}{opt(I)} | I \textit{ is an instance of } P \textit{ with } opt(I) \geq s \right\} \tag{7}$$

is called the asymptotic worst case performance *of L.*

The easiest lower bound for the BPP is the total volume of all items (the size of the container will be assumed to be 1), rounded up to the next integer. For a normalized BPP instance $I := (x_1, \ldots, x_n)$, this bound can be formulated as

$$L_1(I) := \left\lceil \sum_{i=1}^{n} x_i \right\rceil . \tag{8}$$

Theorem 4 (Martello and Toth). $r(L_1) = \frac{1}{2}$.

Martello and Toth showed that for their ILP formulation of the BPP, the bound L_1 dominates the LP relaxation, the surrogate relaxation, and the Lagrange relaxation. (See [16], pp. 224.) According to their numerical experiments, L_1 approximates the optimal value very well, as long as there are sufficiently many items of small size, meaning that the remaining capacity of bins with big items can be exploited. If this is not the case, then the ratio between L_1 and the optimal value can reach a worst case performance of $r(L_1) = \frac{1}{2}$, as shown by the class of examples with $2k$ items of size $\frac{1}{2} + \frac{1}{2k}$.

In the situation which is critical for L_1 ("not enough small items to fill up bins"), we can make use of the dual feasible functions $U^{(\epsilon)}$ from Theorem 2; these functions neglect small items in favor of big ones. This yields the bound L_2: For a BPP instance I, let

$$L_2(I) := \max_{\epsilon \in [0, \frac{1}{2}]} L_1(U^{(\epsilon)}(I)). \tag{9}$$

From Lemma 1, it follows immediately that L_2 is a lower bound for the BPP. The worst case performance is improved significantly:

Theorem 5 (Martello and Toth). $r(L_2) = \frac{2}{3}$.

This bound of $\frac{2}{3}$ is tight: consider the class of BPP instances with $6k$ items of size $\frac{1}{3} + \frac{1}{6k}$.

There is little hope that a lower bound for the BPP that can be computed in polynomial time can reach an absolute worst case performance above $\frac{2}{3}$. Otherwise, the $\mathcal{NP}$-hard problem *Partition* (see [9], p. 47) of deciding whether two sets of items can be split into two sets of equal total size could be solved in polynomial time.

L_2 can be computed in time $O(n \log n)$. The computational effort is determined by sorting the items by their size, the rest can be performed in linear time by using appropriate updates.

Lemma 2 (Martello and Toth). *Consider a BPP instance $I := (x_1, \ldots, x_n)$. If the sizes x_i are sorted, then $L_2(I)$ can be computed in time $O(n)$.*

Now we define for any $k \in \mathbb{N}$

$$L_2^{(k)}(I) := \max_{\epsilon \in [0, \frac{1}{2}]} L_1(u^{(k)} \circ U^{(\epsilon)}(I)) \tag{10}$$

and for $q \geq 2$ consider the following bounds:

$$L_*^{(q)}(I) := \max\{L_2(I), \max_{k=2,\dots,q} L_2^{(k)}(I)\}. \tag{11}$$

By Lemma 1, these are valid lower bounds. As for L_2, the time needed for computing these bounds is dominated by sorting:

Lemma 3. *Let $I := (x_1, \dots, x_n)$ be an instance of the BPP. If the items x_i are given sorted by size, then $L_*^{(q)}(I)$ can be computed in time $O(n)$ for any fixed $q \geq 2$.*

3.2 An Optimality Result for $L_*^{(q)}$

For the described class of worst case instances for L_2, the bound $L_*^{(2)}$ yields the optimal value. As a matter of fact, we have:

Theorem 6. *Let $I := (x_1, \dots, x_n)$ be a BPP instance with all items larger than $\frac{1}{3}$. Then $L_*^{(2)}(I)$ equals the optimal value $opt(I)$.*

Proof. Without loss of generality, let $x_1 \geq x_2 \geq \ldots \geq x_n$. Consider an optimal solution, i. e., an assignment of all items to $m := opt(I)$ bins. Obviously, any bin contains one or two items. In each bin, we call the larger item the *lower* item, while (in the bins with two items) the smaller item is called the *upper* item. In the following three steps, we transform the optimal solution into normal form.

1. Sort the bins in decreasing order of the lower item.
2. Move the upper items into the bins with highest indices. The capacity constraint remains valid, since increasing bin index corresponds to decreasing size of the lower item.
3. Using appropriate swaps, make sure that in bins with two items, increasing index corresponds to increasing size of the upper item. Again, the order of the bins guarantees that no capacity constraint is violated.

Eventually, we get the normal form shown in Figure 4: Item x_i is placed in bin i. The remaining items $m+1, m+2, \dots, n$ are placed in bins $m, m-1, \dots, m-(n-(m+1))$. The first bin with two items has index $q := 2m+1-n$.

For $i \in \{1, \dots, n\}$, the assumption $x_i > \frac{1}{3}$ implies

$$u^{(2)}(x_i) = \frac{\lceil 3x_i - 1 \rceil}{2} \geq \frac{1}{2}. \tag{12}$$

For $n \geq 2m-1$, we get

$$L_2^{(2)}(I) \geq L_1(u^{(2)}(I)) = \left\lceil \sum_{i=1}^{n} u^{(2)}(x_i) \right\rceil \geq \left\lceil \frac{(2m-1)}{2} \right\rceil = m. \tag{13}$$

Hence, the lower bound $L_2^{(2)}(I)$ yields the optimal value, leaving the case

$$n < 2m - 1. \tag{14}$$

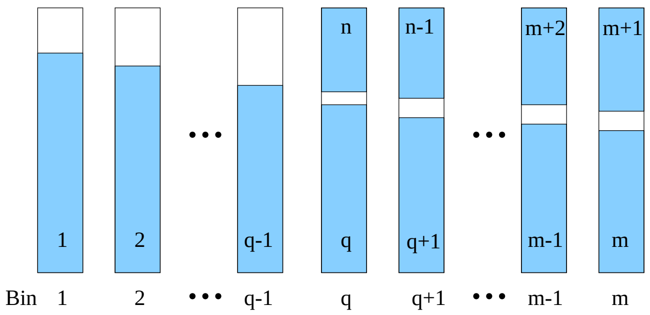

Fig. 4. Normal form of an optimal solution for a BPP instance with sizes $x_i > 1/3$.

Then $q = 2m + 1 - n > 2m + 1 - (2m - 1) = 2$, implying that at least bins 1 and 2 contain only one item.

If $x_m > \frac{1}{2}$, then $x_i > \frac{1}{2}$ holds for all $i \le m$, as the items are sorted by decreasing size, thus $U^{(1/2)}(x_i) = 1$. Hence,

$$L_2(I) \ge L_1(U^{(1/2)}(I)) \ge \left\lceil \sum_{i=1}^{m} U^{(1/2)}(x_i) \right\rceil = m. \tag{15}$$

In this case, $L_2(I)$ equals the optimal value. Therefore, assume

$$x_m \le \frac{1}{2} \tag{16}$$

for the rest of the proof.

If $x_{m-1} + x_m > 1$, let $\epsilon := x_m \le 1/2$. For $i \le m - 1$, we have $x_i \ge x_{m-1} > 1 - x_m = 1 - \epsilon$, hence

$$u^{(2)} \circ U^{(\epsilon)}(x_i) = u^{(2)}(1) = 1. \tag{17}$$

Furthermore, $1/3 < x_m \le 1/2$ implies

$$u^{(2)} \circ U^{(\epsilon)}(x_m) = u^{(2)}(x_m) = \frac{1}{2}. \tag{18}$$

All in all, we have

$$L_2^{(2)}(I) \ge \left\lceil \sum_{i=1}^{m} u^{(2)} \circ U^{(\epsilon)}(x_i) \right\rceil = \left\lceil (m-1) + \frac{1}{2} \right\rceil = m, \tag{19}$$

i. e., the bound meets the optimum.

This leaves the case

$$x_{m-1} + x_m \le 1. \tag{20}$$

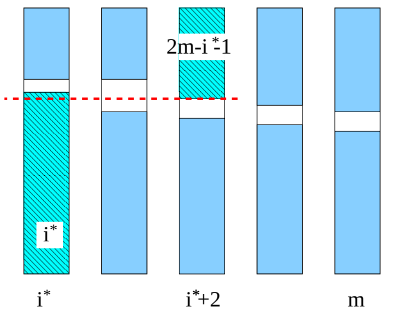

Fig. 5. Determining i^*.

Using the assumptions (14) and (20), we show that there is an $i^* \in \{2m - n, \ldots, n\}$ with

$$x_{i^*} + x_{2m-i^*-1} > 1. \tag{21}$$

Figure 5 shows the meaning of this statement: At least one of the upper items cannot be combined with the lower item two bins to the left.

This is shown in the following way: If for all $i^* \in \{2m - n, \ldots, n\}$, we had

$$x_i + x_{2m-i-1} \leq 1, \tag{22}$$

then all upper items could be moved two bins to the left, since the first two items do not contain more than one item by (14). This would allow it to pack item x_{m-1} with x_m, saving a bin.

Now let $\epsilon := x_{2m-i^*-1}$. By (21), we have for all $i \in \{1, \ldots, i^*\}$ that

$$x_i \geq x_{i^*} > 1 - \epsilon, \tag{23}$$

hence

$$u^{(2)} \circ U^{(\epsilon)}(x_i) = u^{(2)}(1) = 1. \tag{24}$$

For $i \in \{i^* + 1, \ldots, 2m - 1 - i^*\}$ we have

$$x_i \geq x_{2m-i^*-1} = \epsilon, \tag{25}$$

and therefore

$$u^{(2)} \circ U^{(\epsilon)}(x_i) \geq u^{(2)}(x_i) \geq \frac{1}{2}. \tag{26}$$

Summarizing, we get

$$L_2^{(2)}(I) \geq \left\lceil \sum_{i=1}^{i^*} u^{(2)} \circ U^{(\epsilon)}(x_i) + \sum_{i=i^*+1}^{2m-i^*-1} u^{(2)} \circ U^{(\epsilon)}(x_i) \right\rceil \tag{27}$$

$$\geq \left\lceil \sum_{i=1}^{i^*} 1 + \sum_{i=i^*+1}^{2m-i^*-1} \frac{1}{2} \right\rceil \tag{28}$$

$$= \left\lceil i^* + \frac{(2m-2i^*-1)}{2} \right\rceil = m. \tag{29}$$

This completes the proof. □

3.3 Worst Case Performance of $L_*^{(q)}$

As we have stated above, we cannot hope to improve the worst case performance of L_2. However, we can show that the asymptotic worst case performance is improved by $r_\infty(L_*^{(2)})$:

Theorem 7.

$$r_\infty(L_*^{(2)}) = \frac{3}{4}. \tag{30}$$

Proof. Let $I := (x_1, \ldots, x_n)$ be a BPP instance. We start by showing

$$\max\{L_2(I), L_2^{(2)}(I)\} \geq \frac{3}{4} opt(I) - 1. \tag{31}$$

By Theorem 6, all items with $x_i > \frac{1}{3}$ fit into $m \leq L_2^{(2)}(I)$ bins. Let these bins be indexed by $1, \ldots, m$. Using the *First Fit Decreasing* heuristic, we add the remaining items, i. e., sort these items by decreasing order and put each item into the first bin with sufficient capacity; if necessary, use a new bin. Let q denote the number of bins in this solution.

Suppose we need m bins for the big items and not more than $\frac{m}{3}$ items for the rest, then Theorem 6 yields the first part of the statement:

$$opt(I) \leq q \leq \frac{4}{3} m \leq \frac{4}{3} \max\{L_2^{(2)}(I), L_2(I)\}. \tag{32}$$

Therefore, assume

$$\frac{3q}{4} > m \tag{33}$$

for the rest of the proof.

Let α denote the largest free capacity of one of the bins 1 through m, i. e., the total size of all items in this bin is at least $(1-\alpha)$.

No item that was placed in one of the bins $m+1$ through q can fit into the bin with free capacity α. This means that all these bins can only contain items $x_i > \alpha$. On the other hand, the size of these items does not exceed $\frac{1}{3}$. This implies that the bins $m+1$ through $q-1$ must contain at least three items of size $x_i > \alpha$, while bin q holds at least one item $x_i > \alpha$.

Thus, we get

$$L_2(I) \geq L_1(I) = \left\lceil \sum_{i=1}^{n} x_i \right\rceil \geq \lceil (1-\alpha)m + 3\alpha(q-1-m) + \alpha \rceil \tag{34}$$
$$= \lceil (1-4\alpha)m + 3\alpha(q-1) + \alpha \rceil .$$

Now consider two cases.

Case 1: Assume $\alpha > \frac{1}{4}$. Since $(1-4\alpha) < 0$, we can replace the term m in (34) by $\frac{3q}{4} > m$:

$$L_2(I) > \left\lceil (1-4\alpha)\frac{3}{4}q + 3\alpha(q-1) + \alpha \right\rceil \tag{35}$$
$$= \left\lceil \frac{3}{4}q - 2\alpha \right\rceil \tag{36}$$
$$\geq \frac{3}{4}q - 1 \geq \frac{3}{4}opt(I) - 1. \tag{37}$$

Case 2: For $\alpha \leq \frac{1}{4}$, neglect the term $(1-4\alpha)m \geq 0$ in (34):

$$L_2(I) \geq \lceil 3\alpha(q-1) + \alpha \rceil \tag{38}$$
$$\geq \frac{3}{4}q - 1 \geq \frac{3}{4}opt(I) - 1. \tag{39}$$

This proves $r_\infty(L_*^{(2)}) \geq \frac{3}{4}$. For showing that equality holds, consider the family of bin packing instances with $3k$ items of size $1/4+\delta$ with $\delta > 0$. This needs at least k bins. For sufficiently small δ, we have $L_2^{(2)} = 0$ and $L_2(I) = L_1(I) \leq \frac{3}{4}k+1$. □

3.4 Computational Performance of $L_*^{(q)}$

Generally speaking, L_2 yields results that are orders of magnitude better than the worst case performance (see [15,16]). In the following, we compare the computational performance of $L_*^{(q)}$ on the same type of benchmark problems. As it turns out, we get a clear improvement.

For our computational investigation, we generated random instances in the same way as described in [16], pp. 240. The bin size is normalized to 100. For a given number n of items, the sizes were generated randomly with even distribution on the sets $\{1,\ldots,100\}$, $\{20,\ldots,100\}$, and $\{35,\ldots,100\}$. For each problem class, and $n \in \{100, 500, 1000\}$, we generated 1000 instances. In the table, we compare L_2 with the bounds $L_*^{(2)}$, $L_*^{(5)}$, and $L_*^{(10)}$. Shown is the average relative error in percent (% err) and the number of instances, for which the optimal value was reached (# opt). The optimum was computed with the routine MTP from [16] with a limit of 100000 search nodes.

Especially for large instances, we see a clear improvement by the new bounds. Compared with L_2, the number of times that the optimal value is met is increased by 50 % for $n = 1000$ and the first two problem classes. For the third problem class, we always get the optimum, as shown in Theorem 6. The differences between $L_*^{(2)}$, $L_*^{(5)}$ and $L_*^{(10)}$ are significant, but small.

Table 1. Performance of lower bounds for the BPP.

		L_2		$L_*^{(2)}$		$L_*^{(5)}$		$L_*^{(10)}$	
Interval	n	% err	# opt	% err	# opt	% err	# opt	% err	# opt
$[1, 100]$	100	0.432	774	0.324	832	0.303	843	0.272	859
	500	0.252	474	0.157	644	0.154	645	0.130	693
	1000	0.185	381	0.116	578	0.114	578	0.100	606
$[20, 100]$	100	0.419	732	0.297	812	0.265	832	0.232	853
	500	0.231	443	0.144	634	0.138	642	0.123	677
	1000	0.181	366	0.104	605	0.103	605	0.091	632
$[35, 100]$	100	0.229	827	0.000	1000	0.000	1000	0.000	1000
	500	0.160	553	0.000	1000	0.000	1000	0.000	1000
	1000	0.114	507	0.000	1000	0.000	1000	0.000	1000

4 Conclusions

We have presented a fast new method for generating lower bounds for the bin packing problem. The underlying method of dual feasible functions can also be used in the case of higher dimensions by combining our ideas the approach for modeling higher-dimensional orthogonal packings that we developed for finding exact solutions for the d-dimensional knapsack problem [4]. Details will be contained in the forthcoming papers [5,6,7].

References

1. J. E. Beasley. OR-Library: Distributing test problems by electronic mail. *Journal of Operations Research Society*, 41:1069–1072, 1990. `http://mscmga.ms.ic.ac.uk/info.html`
2. E. G. Coffmann, Jr., M. R. Garey, and D. S. Johnson. Approximation algorithms for bin packing: A survey. In D. S. Hochbaum, editor, *Approximation Algorithms for NP-Hard Problems*, pages 46–93. PWS Publishing, Boston, 1997.
3. E. G. Coffmann, Jr. and G. S. Lueker. *Probabilistic Analysis of Packing and Partitioning Algorithms.* Wiley, New York, 1991.
4. S. P. Fekete and J. Schepers. A new exact algorithm for general orthogonal d-dimensional knapsack problems. *Algorithms – ESA '97, LNCS, Vol. 1284*, pages 144–156. Springer, 1997.
5. S. P. Fekete and J. Schepers. On more-dimensional packing I: Modeling. ZPR Technical Report 97-288. `http://www.zpr.uni-koeln.de`
6. S. P. Fekete and J. Schepers. On more-dimensional packing II: Bounds. ZPR Technical Report 97-289. `http://www.zpr.uni-koeln.de`
7. S. P. Fekete and J. Schepers. On more-dimensional packing III: Exact Algorithms. ZPR Technical Report 97-290. `http://www.zpr.uni-koeln.de`
8. W. Fernandez de la Vega and G. S. Lueker. Bin packing can be solved within $1 + \varepsilon$ in linear time. *Combinatorica*, 1:349–355, 1981.
9. M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* Freeman, San Francisco, 1979.

10. I. P. Gent. Heuristic solution of open bin packing problems. (To appear in *Journal of Heuristics*). `http://www.cs.strath.ac.uk/~apes/papers`
11. D. S. Johnson. *Near-optimal bin packing algorithms.* PhD thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, 1973.
12. N. Karmarkar and R. M. Karp. An efficient approximation scheme for the one-dimensional bin packing problem. *Proc. 23rd Annual Symp. Found. Comp. Sci. (FOCS 1982)*, pages 312–320, 1982.
13. G. S. Lueker. Bin packing with items uniformly distributed over intervals $[a, b]$. *Proc. 24th Annual Symp. Found. Comp. Sci. (FOCS 1983)*, pages 289–297, 1983.
14. S. Martello, D. Pisinger, and D. Vigo. The three-dimensional bin packing problem. Technical Report DEIS-OR-97-6, 1997. `http://www.deis.unibo.it`
15. S. Martello and P. Toth. Lower bounds and reduction procedures for the bin packing problem. *Discrete Applied Mathematics*, 28:59–70, 1990.
16. S. Martello and P. Toth. *Knapsack Problems.* Wiley, New York, 1990.
17. S. Martello and D. Vigo. Exact solution of the two-dimensional finite bin packing problem. Technical Report DEIS-OR-96-3, 1996. `http://www. deis.unibo.it`
18. J. Schepers. *Exakte Algorithmen für Orthogonale Packungsprobleme.* PhD thesis, Mathematisches Institut, Universität zu Köln, 1997.