Maximizing the number of unused bins

Marc Demange, Jérôme Monnot, Vangelis Th. Paschos

MAXIMIZING THE NUMBER OF UNUSED BINS

Marc Demange* Jérôme Monnot Vangelis Th. Paschos

LAMSADE, Université Paris-Dauphine
Place du Maréchal De Lattre de Tassigny, 75775 Paris Cedex 16, France
e_mails: {demange,monnot,paschos}@lamsade.dauphine.fr

Abstract. We analyze the approximation behavior of some of the best-known polynomial-time approximation algorithms for *bin-packing* under an approximation criterion, called *differential ratio*, informally the ratio $(n - \lambda(I))/(n - \beta(I))$, where n is the size of the input list, $\lambda(I)$ is the size of the solution provided by an approximation algorithm and $\beta(I)$ is the size of the optimal one. This measure has originally been introduced by Ausiello, D'Atri and Protasi and more recently revisited, in a more systematic way, by the first and the third authors of the present paper. Under the differential ratio, bin-packing has a natural formulation as the problem of maximizing the number of unused bins. We first show that two basic fit bin-packing algorithms, the first-fit and the best-fit, admit differential approximation ratios 1/2. Next, we show that slightly improved versions of them achieve ratios 2/3. Refining our analysis we show that the famous first-fit-decreasing and best-fit decreasing algorithms achieve differential approximation ratio 3/4. Finally, we show that first-fit-decreasing achieves asymptotic differential approximation ratio 7/9.

1 Introduction

An **NP** optimization problem Π is defined as a quintuple $(\mathcal{I}, \mathsf{S}, v_I, S^t, \text{opt})$ such that:

- $\mathcal{I}$ is the set of instances of Π and it can be recognized in polynomial time;

*Also, CERMSEM, Université Paris I, Maison des Sciences Economiques, 106-112 boulevard de l'Hôpital, 75647 Paris Cedex 13, France

- given $I \in \mathcal{I}$ (let $|I|$ be the size of I), $\mathsf{S}(I)$ denotes the set of feasible solutions of I; moreover, there exists a polynomial P such that, for every $S \in \mathsf{S}(I)$ (let $|S|$ be the size of S), $|S| = O(P(|I|))$; furthermore, given any I and any S with $|S| = O(P(|I|))$, one can decide in polynomial time if $S \in \mathsf{S}(I)$;
- given $I \in \mathcal{I}$ and $S \in \mathsf{S}(I)$, $v_I(S)$ denotes the value of S; v_I is polynomially computable and is commonly called objective function;
- given $I \in \mathcal{I}$, $\exists S^t \in \mathsf{S}(I)$ computable in polynomial time;
- opt $\in \{\max, \min\}$.

The most interesting sub-class of **NP** optimization problems is the class of the **NP**-hard ones, known to be unsolvable in polynomial time unless **P=NP** and people widely thinks that this fact is very unlikely.

A current and very active research area coping with **NP**-hardness is *polynomial-time approximation theory*. In this domain, the main objective is either finding a good approximation algorithm for a given **NP**-hard problem, or establishing proofs that such algorithms cannot exist unless an unlikely complexity-theory condition (for example, **P=NP**) holds. The "goodness" of an approximation algorithm is commonly measured by its approximation ratio.

Given an instance I of a combinatorial optimization problem Π and an approximation algorithm $\mathtt{A}$ supposed to feasibly solve Π, we will denote by $\omega(I)$, $\lambda_{\mathtt{A}}(I)$ and $\beta(I)$ the values of the worst solution, the approximated one (provided by $\mathtt{A}$), and the optimal one, respectively.

There mainly exist two ways of dealing with polynomial-time approximation. Traditionally ([5, 7]), the quality of an approximation algorithm for an **NP**-hard minimization (resp., maximization) problem Π is expressed by the ratio (called standard in what follows) $\rho_{\mathtt{A}}(I) = \lambda_{\mathtt{A}}(I)/\beta(I)$, and the quantity $\rho_{\mathtt{A}} = \inf\{r : \rho_{\mathtt{A}}(I) < r, I \text{ instance of } \Pi\}$ (resp., $\rho_{\mathtt{A}} = \sup\{r : \rho_{\mathtt{A}}(I) > r, I \text{ instance of } \Pi\}$) constitutes the standard approximation ratio of $\mathtt{A}$ for Π. Recent works ([3, 2]), strongly inspired by former ones (see, for example, [1]), bring to the fore another approximation measure, as powerful as the traditional one (concerning the type, the diversity and the quantity of the produced results), the ratio (called differential in what follows) $\delta_{\mathtt{A}}(I) = (\omega(I) - \lambda_{\mathtt{A}}(I))/(\omega(I) - \beta(I))$. The quantity $\delta_{\mathtt{A}} = \sup\{r : \delta_{\mathtt{A}}(I) > r, I \text{ instance of } \Pi\}$ is now the differential approximation ratio of $\mathtt{A}$ for Π.

Starting from approximation ratio, one can rule out a set of instances, for example instances for which the optimal solution value is bounded, restraining so itself/herself to instances with unbounded optimal values. Then, one can try to evaluate the approximation performance of an algorithm over these instances so studying what in approximation theory is called asymptotic approximation ratio. This ratio in standard approximation framework is defined, for minimization problems, by $\rho^{\infty}_{\mathtt{A}} = \inf\{\rho \geq 1 : \exists \kappa \in \mathbf{Z}^+, \rho_{\mathtt{A}}(I) \leq \rho, \forall I : \beta(I) \geq \kappa\}$ ([5]).

On the other hand, as one can easily see, a possible interpretation of the differential approximation ratio is that it expresses the position of $\lambda(I)$ into the interval of the feasible objective function's values (as we have restricted ourselves to discrete problems, there exists a finite number of such values). Moreover, for most

of the maximization problems, since the value of the worst solution is equal to 0, the optimal value is an upper bound for the number of the feasible values. Such observations provide the terms of another parameter following the value of which one can rule out a set of instances, *the number $\sigma(I)$ of the feasible values of the solutions of I*. Based upon this parameter, we define the *asymptotic differential approximation ratio* of a polynomial-time approximation algorithm $\mathtt{A}$ as

$$\delta_{\mathtt{A}}^{\infty} = \lim_{k\to\infty} \inf_{\substack{I \\ \sigma(I)\geq k}} \left\{ \frac{\omega(I) - \lambda_{\mathtt{A}}(I)}{\omega(I) - \beta(I)} \right\}.$$

In other words, $\delta_{\mathtt{A}}^{\infty}$ is the lower limit of the sequence of ratios indexed by the value of $\sigma(I)$ (more details and further motivations about this asymptotic ratio are given in [4]). Let us note that the condition $\sigma(I) \geq k$ used in the definition of the asymptotic approximation ratio for characterizing "the sequence of unbounded instances" (or "limit instances") cannot be polynomially verified[1]. But in practice, for a given problem, it is possible to directly interpret condition $\sigma(I) \geq k$ by means of $\omega(I)$ and $\beta(I)$. For example, for numerous cases of discrete problems, we are able to determine, for each instance, a step π defined as the least variation between two feasible values of the instance (for example, for bin-packing $\pi = 1$); then, $\sigma(I) \leq ((\omega(I) - \beta(I))/\pi) + 1$ and consequently,

$$\delta_{\mathtt{A}}^{\infty} = \lim_{k\to\infty} \inf_{\substack{I \\ \sigma(I)\geq k}} \left\{ \frac{\omega(I) - \lambda_{\mathtt{A}}(I)}{\omega(I) - \beta(I)} \right\} \geq \lim_{k\to\infty} \inf_{\substack{I \\ \frac{\omega(I)-\beta(I)}{\pi}\geq k-1}} \left\{ \frac{\omega(I) - \lambda_{\mathtt{A}}(I)}{\omega(I) - \beta(I)} \right\}.$$

Whenever π can be determined, condition $(\omega(I) - \beta(I))/\pi \geq k - 1$ can be more easily evaluated than $\sigma(I) \geq k$, and in this case, the former is used (this is not senseless since we try to bound below the ratio).

As it is shown in [3, 2], many problems as, for example, minimum graph-coloring, minimum vertex-covering, etc., behave in completely different ways regarding traditional or differential approximation.

2 About bin-packing

In the bin-packing problem (BP) we are given a finite set $L = \{x_1, \ldots, x_n\}$ of n rational numbers and an unbounded number of bins (denoted by $b_1, b_2, \ldots$in what follows) of capacity 1; we wish to arrange all these elements in the least possible bins in such a way that the sum of the numbers in each bin does not violate its capacity. BP is **NP**-hard and, consequently, no polynomial-time algorithm can exactly solve it, unless **P=NP**. Remark that the worst BP-solution for L consists in

[1] Of course, the same holds for the condition $\beta(I) \geq k$ in standard approximation framework.

taking a bin per item. So, the differential ratio of an approximation BP-algorithm $\mathtt{A}$ equals $(n - \lambda_{\mathtt{A}}(L))/(n - \beta(L))$. Moreover, remark that the least the bins used by an algorithm in order to arrange the elements of L, the most the bins unused (if we consider that, at worst, no more than n bins will ever used). In this sense, solving BP in the differential framework, is equivalent to solving a maximization problem in the standard framework, this maximization problem consisting of maximizing the number of unused bins. Consequently, the use of the differential measure provides for BP an insight which cannot be derived from the standard one.

It is easy to see that for BP its subproblem where input-lists verify $\sigma(I) \leq k$ for a fixed $k \in I\!N$ is polynomial ($O(n^k)$). So, the following lemma immediately holds.

Lemma 1. *For every polynomial time approximation algorithm $\mathtt{A}$ and for all $\epsilon > 0$, there exists a polynomial time approximation algorithm $\mathtt{A}_\epsilon$ for which $\delta_{\mathtt{A}_\epsilon} \geq \delta^{\infty}_{\mathtt{A}} - \epsilon$.*

Let us note that the above lemma does not hold for every **NP** optimization problem, for example, for the bottleneck ones.

The purpose of this paper is to study the differential approximation behavior of some very popular BP-algorithms. We first analyze two simple and intuitive ones, the first-fit (FF) and the best-fit (BF). Next, we study one of the most-known BP-algorithm, the first-fit-decreasing one (FFD) where, as we point out later, is a refinement of FF.

Crucial notion in our approximation analysis is the one of *n-worst instance*. Given an **NP**-hard problem Π and an approximation algorithm $\mathtt{A}$ for Π, the set of n-worst instances defines a set of instances in which $\mathtt{A}$ performs the worst possible with respect to the differential approximation ratio (in other words, the set of the n-worst instances consists of the instances in which $\delta_{\mathtt{A}}$ is the smallest over all instances of size at most n). So, for a pair $(\Pi, \mathtt{A})$, evaluation of $\delta_{\mathtt{A}}$ in such an instance of Π immediately provides a lower bound for $\delta_{\mathtt{A}}$ on every Π-instance of a certain size. For the case of BP, an n-worst instance can be defined as follows.

Definition 1. Consider a size n and an approximation algorithm $\mathtt{A}$ for BP. Then, an *n-worst instance* for BP with respect to $\mathtt{A}$ is a list $L^w \in \mathrm{argmin}_{|L| \leq n}\{\delta_{\mathtt{A}}(L)\}$. An n-worst instance for $(\mathrm{BP}, \mathtt{A})$ of minimum size will be called *minimal n-worst instance*. ∎

Let us remark that an n-worst instance L^w for BP exists for every n, since the set $\{\delta_{\mathtt{A}}(L) : |L| \leq n\}$ is finite. If n is sufficiently large, then we have $\delta_{\mathtt{A}}(L^w) < 1$. In general, the study of the approximation behavior of $\mathtt{A}$ in any family of BP-instances containing n-worst instances (with n arbitrarily large) suffices to provide a lower bound for $\delta_{\mathtt{A}}$.

3 First-fit, best-fit and preliminary results

In what follows in sections 3 and 4, given a list L of elements, we denote by x_1, x_2 and x_3, $x_1 \leq x_2 \leq x_3$, the three smallest elements of L and by x_n the larger element of L. Moreover, through the paper a bin i will be denoted either by b_i, or by the set of its elements; a BP-solution will be alternatively denoted by the union of its bins.

The spirits of algorithms FF and BF are quite similar; they only differ by the way of choosing the bin in which they will place the first unplaced element. We suppose that they start with a set of n initially empty bins. For reasons of economy we present in parallel the two algorithms; instructions between starry parentheses refer to BF.

```
BEGIN /*FF(*BF*)*/
      IF L ≠ ∅ THEN j ← min{k : Σ_{x1∈bk} x ≤ 1 − x1};
                    (*b ← argmax{Σ_{x∈bk} x : Σ_{x∈bk} x ≤ 1 − x1};*)
                    bj ← bj ∪ {x1};
                    (*b ← b ∪ {x1};*)
                    FF(L \ {x1});
                    (*BF(L \ {x1});*)
      FI
      OUTPUT the set FFL(*BFL*) of non empty bins;
END. /*FF(*BF*)*/
```

It is easy to see that both FF and BF work in polynomial time.

Let A be one of FF and BF; the following lemma 2 points out a very simple property of BP and of A, called Bellman-like principle. Roughly speaking, given a BP-instance L and an optimal (resp., A-) solution for L, if one removes some of its bins and if he/she considers the sub-instance L' consisting of the content of the remaining bins, then the surviving set of bins remains an optimal (resp., feasible for A-) solution with respect to L'. More precisely, the following lemma holds.

Lemma 2. *Let L be an instance of BP and denote by $(b_j^*, j = 1, \ldots, \beta(L))$ (resp., $(b_j, j = 1, \ldots, \lambda_{\mathtt{A}}(L))$ an optimal (resp., the A-) solution of instance L. Then,*

- *for every subset $J \subset \{1, \ldots, \beta(L)\}$, the solution $(b_j^*, j \in J)$ is optimal for instance $\cup_{j \in J} b_j^*$;*
- *similarly, for every subset $I \subset \{1, \ldots, \lambda_{\mathtt{A}}(L)\}$, A computes, in instance $\cup_{i \in I} b_i$, the solution $(b_i, i \in I)$.*

The proof of the results of this paper is substantially based upon the following lemma 3, exploiting the notion of n-worst instance.

Lemma 3. *Let A be one of FF and BF, let $n \geq 0$, let L be an n-worst instance for $(BP, \mathtt{A})$ and let $b_1, \ldots, b_{\lambda_{\mathtt{A}}(L)}$ ($b_i \neq \emptyset$) be the solution provided by A. If there*

exists a set of bins $b_i, i \in I, I \subset \{1, \ldots, \lambda_{\mathtt{A}}(L)\}$ *such that the list* $L' = L \setminus \cup_{i \in I} b_i$ *satisfies* $\beta(L') \leq \beta(L) - x$*, with* $0 \leq x \leq y \leq z$*,* $x \neq z$*, where* $y = |I|$ *and* $z = |\cup_{i \in I} b_i|$*, then* $\delta_{\mathtt{A}}(L) \geq (z - y)/(z - x)$*.*

Proof. Let us first point out that the result is valid if $L' = \emptyset$ because, in this case, the hypotheses of the lemma imply $\lambda_{\mathtt{A}}(L) = y$ and $\beta(L) \geq x$. So we can consider $L' \neq \emptyset$ and the following holds.

$$\lambda_{\mathtt{A}}(L') = \lambda_{\mathtt{A}}(L) - y \quad (1)$$
$$\beta(L') \leq \beta(L) - x \quad (2)$$
$$\omega(L') = \omega(L) - z \quad (3)$$

Where expression (3) holds because the size of instance L' is $n' = |L| - z \geq 0$, expression (1) holds thanks to lemma 2 and expression (2) is one of the hypotheses of the lemma.

Revisit now definition 1 and recall that L is an n-worst instance; this implies in particular that $\delta_{\mathtt{A}}(L) \leq \delta_{\mathtt{A}}(L')$; moreover suppose $D = \omega(L) - \beta(L) - (z - x) > 0$. Then, expressions (1), (2) and (3) lead to

$$\frac{\omega(L) - \lambda_{\mathtt{A}}(L)}{\omega(L) - \beta(L)} \leq \frac{\omega(L') - \lambda_{\mathtt{A}}(L')}{\omega(L') - \beta(L')} \leq \frac{\omega(L) - \lambda_{\mathtt{A}}(L) - (z - y)}{\omega(L) - \beta(L) - (z - x)};$$

this, after some easy algebra, implies $\delta_{\mathtt{A}}(L) = (\omega(L) - \lambda_{\mathtt{A}}(L))/(\omega(L) - \beta(L)) \geq (z - y)/(z - x)$.

Let us now suppose that $D \leq 0$. Then, $\omega(L) - \beta(L) \leq z - x$ and, since from expressions (1) and (3), $\omega(L) - \lambda_{\mathtt{A}}(L) = \omega(L') - \lambda_{\mathtt{A}}(L') + (z - y) \geq z - y$, then immediately $\delta_{\mathtt{A}}(L) = (\omega(L) - \lambda_{\mathtt{A}}(L))/(\omega(L) - \beta(L)) \geq (z - y)/(z - x)$, q.e.d. ∎

For reasons of concision of the proofs of the results, we consider x, y and z appearing in the above lemma 3 as parameters of the lemma; so, we will speak about (x, y, z)*-application of lemma 3.*

Theorem 1. $\delta_{\mathtt{FF}} \geq 1/2$*; moreover, bound 1/2 is tight.*

Proof. We will prove that the approximation ratio of FF, when executed on n-worst instances, is always at least $1/2$. Let L be such an n-worst instance for the pair $(\mathrm{BP}, \mathtt{FF})$.

If $\lambda_{\mathtt{FF}}(L) = \omega(L) = n$ (i.e., FF uses a bin per element), then $\lambda_{\mathtt{FF}}(L) = \beta(L)$. In fact, $\lambda_{\mathtt{FF}}(L) = \omega(L) = n$ implies $x_1 + x_2 > 1$ (recall that x_1 and x_2 are the two smallest elements of L) because, if not, FF would place x_1 and x_2 in the same bin. So, for every pair (x, y) of elements in L, $x + y > 1$. In this case, in every solution for L, every element will be arranged in its own bin; so, $\lambda_{\mathtt{FF}}(L) = \beta(L) = \omega(L) = n$. Moreover, this can never be the case for an n-worst instance for $(\mathrm{BP}, \mathtt{FF})$ for a sufficiently large n (given that, by definition, in such an instance FF realizes its worst approximation performance), since then $\mathbf{P} = \mathbf{NP}$.

Hence, let us consider that there exists at least a bin $b \in \mathrm{FFL}$ containing at least two elements and denote by $L' = (\cup_{b_i \in \mathrm{FFL}} b_i) \setminus b$. List L' meets conditions of lemma 3 with $(x, y, z) = (0, 1, 2)$. So, (0,1,2)-application of lemma 3 gets

$\delta_{\mathtt{FF}}(L) = (\omega(L) - \lambda_{\mathtt{FF}}(L))/(\omega(L) - \beta(L)) \geq 1/2$. Since L is an n-worst instance, the bound $1/2$ obtained for L is a lower bound for $\delta_{\mathtt{FF}}$.

In order to prove the tightness of the approximation ratio obtained, we will prove a stronger result, namely that the *asymptotic* approximation ratio of FF is bounded above by $1/2$. For this, let us consider a $4k$-element list L containing $2k$ elements equal to $(1/2) - \epsilon$ followed by $2k$ elements equal to $(1/2) + \epsilon$ for some $\epsilon \in (0, 1/2)$. It is easy to see that FF(L) will produce k bins containing all the elements equal to $(1/2) - \epsilon$ (two such elements per bin), and $2k$ bins each one containing a element equal to $(1/2) + \epsilon$; so, $\lambda_{\mathtt{FF}}(L) = 3k$. On the other hand, in the optimal solution, $2k$ bins (each bin containing a pair $((1/2) - \epsilon, (1/2) + \epsilon)$) suffice to arrange all the elements of L. Finally, $\omega(L) = 4k$. Since $\sigma(L) = \omega(L) - \beta(L) + 1 = 2k + 1$ and k can be chosen arbitrarily large, $\delta^{\infty}_{\mathtt{FF}}(L) = 1/2$. It suffices now to note that, for every approximation algorithm A, $\delta^{\infty}_{\mathtt{A}} \geq \delta_{\mathtt{A}}$, and this concludes the proof of the tightness of the bound obtained and of the theorem. ∎

Remark 1. Since $\delta^{\infty}_{\mathtt{FF}} = \delta_{\mathtt{FF}}$, one cannot use FF and lemma 1 in order to devise a polynomial time approximation algorithm with differential approximation ratio better than the one of algorithm FF. ∎

With arguments exactly similar to the ones of theorem 1 (and via the same counter-example), one can prove that algorithm BF also achieves a tight differential approximation ratio $1/2$.

Theorem 2. $\delta_{\mathtt{BF}} \geq 1/2$*; moreover, bound 1/2 is tight.*

4 Improving first-fit and best-fit differential ratios

The proof of the tightness for $\delta_{\mathtt{FF}}$ (theorem 1) has confirmed once more the (very intuitive) fact, largely pointed out by many authors, that the most of the BP-algorithms (all the fit-based ones in any case) work worse in increasingly ordered lists than in unordered or in decreasingly ordered ones. Based upon this phenomenon, we will try to improve differential approximation ratios of FF and BF.

We will now consider a slightly improved version of FF, called FFI. This latter algorithm is, in fact, algorithm FF matched, during its last phase, with OPT which optimally solves the case where the sum of the three smallest elements of the (increasingly ordered) list is at least equal to 1. Both algorithms are recursive. In what follows, we will call *last_bin* the (non-empty) bin containing the last (greater index) placed element of a list L. If no elements have been placed yet, then last_bin $= b_1$. We suppose that all but the first recursive calls of FFI do not execute its first line (i.e., the surviving lists are not reordered; also recall that following the conventions of the beginning of the section, the three first elements of the input-list – and of

the surviving ones – are the three smallest elements of the list and the last element the largest one).

```
BEGIN /*FFI*/
      order L in increasing order;
      IF L ≠ ∅ AND x1 + x2 + x3 ≤ 1 THEN j ← min{k : Σ_{x∈b_k} x ≤ 1 − x1};
                                         b_j ← b_j ∪ {x1};
                                         L ← L \ {x1};
                                         FFI(L);
      FI
      let B be the set of the non-empty bins;
      IF L ≠ ∅ THEN b ← last_bin;
         IF Σ_{x∈b} x + x1 ≤ 1 THEN b ← b ∪ {x1};
                                    L ← L \ {x1};
         FI
         IF Σ_{x∈b} x + x2 ≤ 1 THEN b ← b ∪ {x2};
                                    L ← L \ {x2};
         FI
      FI
      OUTPUT B ∪ {OPT(L) ← {BEGIN /*OPT*/
                                IF x1 + xn ≥ 1 THEN
                                             put x1 in a new bin;
                                             L ← L \ {x1};
                                             OPT(L);
                                             ELSE
                                             put x1, xn in a new bin;
                                             L ← L \ {x1, xn};
                                             OPT(L);
                                FI
                                OUTPUT the set of used bins;
                           END. /*OPT*/}}
END. /*FFI*/
```

It is easy to see that both OPT and FFI work in polynomial time.

Theorem 3. $\delta_{\mathtt{FFI}} \geq 2/3$ *and this bound is tight.*

Proof. Let us consider an n-worst instance L for the pair (BP, FFI).

If $x_1 + x_2 + x_3 \leq 1$, then execution of algorithm FFI on L will arrange at least elements x_1, x_2 and x_3 in the same bin b_1; consequently, the solution returned by FFI will contain at least a bin with three elements. In this case, list $L' = L \setminus \{x_1, x_2, x_3\}$ meets the hypotheses of lemma 3. Consequently, a (0,1,3)-application of this lemma and given that L is n-worst for the pair (BP, FFI), achieves approximation ratio 2/3 for BP.

Let us suppose now that the n-worst instance L on which algorithm FFI is called needs also execution of algorithm OPT (in the final part of L); this is the case where $x_1 + x_2 + x_3 > 1$. In this case, since the sum of the three smallest elements

of L is already greater than 1, no triple of elements of L can be arranged in the same bin; hence, every BP-solution in L even an optimal one will be constituted of bins containing at most two elements. We will prove that in this case execution of `OPT` computes an optimal BP-solution on L. Let us consider such a solution and denote it by B^*. We study the following cases, namely $x_1 + x_n > 1$ (case 1) and $x_1 + x_n \leq 1$ (case 2).

Case 1: $\mathbf{x_1 + x_n > 1}$. Then, since $x_n \geq x_1$, element x_n will occupy a bin by itself in every feasible solution for L; so, solution B^* and the one computed by `OPT(L)` coincide with respect to x_n. This concludes case 1.

Case 2: $\mathbf{x_1 + x_n \leq 1}$. Let us consider the bin containing x_1 in B^*. We then distinguish the following sub-cases depending on whether x_1 occupies a bin by itself (subcase 2.1), or it shares the same bin with another element (subcase 2.2).

Subcase 2.1: $\mathbf{x_1}$ occupies a bin by itself. Then, since $x_1 + x_n \leq 1$, if one removes x_n from the bin of B^* containing it and if one places x_n in the bin containing x_1, then one will produce a feasible solution for L containing no more bins than B^*. It suffices now to remark that algorithm `OPT(L)` (for the case where $x_1 + x_n \leq 1$) arranges x_1 and x_n; so, the solution computed by `OPT(L)` coincides with an optimal solution with respect to $\{x_1, x_n\}$. This concludes subcase 2.1.

Subcase 2.2: $\mathbf{x_1}$ shares the same bin with another element. Suppose that x_k is this element, and that $x_k \neq x_n$ (if not, solution B^* and the one computed by `OPT(L)`, coincide with respect to $\{x_1, x_n\}$). Furthermore, remark that $x_k \leq x_n$. So, one can interchange x_k and x_n in B^* producing so a new feasible solution using the same number of bins as B^* and coinciding with the one computed by `OPT(L)` with respect to $\{x_1, x_n\}$. This concludes subcase 2.2 and case 2.

Consequently, inductive application of cases 1 and 2 in B^* easily shows that it coincides perfectly with the BP-solution computed by `OPT(L)`; hence, `OPT` is optimal when called in ordered (in increasing order) lists, where the sum of the three smallest elements is greater than 1.

Finally, it is easy to conclude that `FFI` achieves differential approximation ratio bounded below by $\min\{2/3, 1\} = 2/3$ and this concludes the proof of the approximation bound. Also, let us note that, for sufficiently large n, an n-worst instance verifies $x_1 + x_2 + x_3 \leq 3$.

As previously in theorem 1, in order to prove the tightness of the above bound, we will prove the stronger claim that the asymptotic approximation ratio of `FFI` is bounded above by 2/3. Let us consider a $6k$-element list L containing $3k$ elements equal to 1/3 followed by $3k$ elements equal to 2/3. Then `FFI(L)` will produce k bins containing all the elements equal to 1/3 (three such elements per bin) and $3k$ bins, each one containing an element equal to 2/3 (one such element per bin); so, $\lambda_{\texttt{FFI}}(L) = 4k$. Obviously, $\omega(L) = 6k$. Finally, optimal solution will use $3k$ bins (each bin containing a pair $(1/3, 2/3)$). Since $\sigma(L) = \omega(L) - \beta(L) + 1 = 3k + 1$ and k can be, once more, chosen arbitrarily large, $\delta^{\infty}_{\texttt{FFI}}(L) = 2/3$. ∎

Arguments similar to the ones proving theorem 3 lead to the following result.

Theorem 4. *Consider algorithm `BFI` which is `FFI` with execution of `FF` substituted by execution of `BF`. Then, $\delta_{\texttt{BFI}} \geq 2/3$ and this ratio is tight.*

Remark 2. The test $L \neq \emptyset$ performed before calling `OPT(L)` in algorithm `FFI` is done in order to fill up last_bin the most possible and to avoid configurations as the following one. Consider a 4-element list $L = \{\epsilon, (1/2) - 2\epsilon, (1/2) + \epsilon, 1 - \epsilon\}$; here, $x_1 + x_2 + x_3 \leq 1$ and $x_2 + x_3 + x_4 > 1$. Then `FF(L)` will create a bin $b_1 = \{x_1 = \epsilon\}$. In the new list $L = \{(1/2) - 2\epsilon, (1/2) + \epsilon, 1 - \epsilon\}$, the second condition of the first `IF`-clause is not verified, consequently no new execution of `FF(L)` is performed. If we proceeded immediately in a call of `OPT(L)`, then a new bin $b_2 = \{x_3 = 1 - \epsilon\}$ would be produced and finally a last bin $b_3 = \{x_1 = (1/2) - 2\epsilon, x_2 = (1/2) + \epsilon\}$ would conclude application of (the poor version of) `FFI`. So, $B = \{b_1\}$, last_bin = b_1, OPT(L) = $\{b_2, b_3\}$; one optimal solution for L is $\{\{\epsilon, (1/2) - 2\epsilon, (1/2) + \epsilon\}, \{1 - \epsilon\}\}$. On the other hand, with the test $L \neq \emptyset$, the solution computed by `FFI` will include two bins, $b_1 = \{\epsilon, (1/2) - 2\epsilon, (1/2) + \epsilon\}$, the first element being included in b_1 thanks to `FF(L)` and the two last elements thanks to the execution of the test. Finally, execution of `OPT(L` $= \{\mathtt{x}_1 = 1 - \epsilon\}$) creates a bin $b_2 = \{1 - \epsilon\}$. ∎

Remark 3. Since $\delta^{\infty}_{\mathtt{FFI}} = \delta_{\mathtt{FFI}}$ and $\delta^{\infty}_{\mathtt{BFI}} = \delta_{\mathtt{BFI}}$, remark 1 holds also for algorithms `FFI` and `BFI` ∎

5 First-fit and best-fit-decreasing

Algorithms `FFD` and `BFD` are further refinements of `FF` and `BF`, respectively, where bins are (arbitrarily) numbered, the input-list is first ordered in decreasing order and next, one puts an element in the smaller order (first) bin which can receive it (without overflow). In what follows, we give a specification of both algorithms in a format analogous to the one used for `FF` and `BF`. For reasons of simplicity, we restrict ourselves to the case of `FFD`; study of `BFD` uses analogous arguments. Moreover, we change our conventions with respect to the previous section and consider that (since `FFD` initially orders input-lists in decreasing order) x_1 is the largest element of L and x_n the smallest one. Finally, we consider as previously that all but the first recursive calls of `FFD` do not execute the first two instructions.

```
BEGIN /*FFD(*BFD*)*/
      order L in decreasing order;
      let L = {x_1, ..., x_n} be the ordered list obtained;
      IF L ≠ ∅ THEN j ← min{k : Σ_{x∈b_k} x ≤ 1 − x_1};
                    (*b ← argmax{Σ_{x∈b_k} x : Σ_{x∈b_k} x ≤ 1 − x_1};*)
                    b_j ← b_j ∪ {x_1};
                    (*b ← b ∪ {x_1};*)
                    FFD(L \ {x_1});
                    (*BFD(L \ {x_1});*)
```

```
      FI
      OUTPUT the list of non-empty bins;
END. /*FFD(*BFD*)*/
```

Since FFD works on decreasingly ordered lists, we restrict our analysis only in such lists $L = (x_1, \ldots, x_n)$ (with $x_1 \geq \ldots \geq x_n$). Moreover, note that lemmata 2 and 3 hold also for FFD.

Given $n \geq 6$, let us now consider $L_0 = (x_1, \ldots, x_{n_0})$ a minimal n-worst instance and denote by n_0 ($n_0 \leq n$) its size; as mentioned above, we assume, without loss of generality, that this list is sorted in decreasing order, (i.e., $x_1 \geq \ldots \geq x_{n_0} > 0$). We denote by $B = (b_1, \ldots, b_{\lambda_{\texttt{FFD}}(L_0)})$ and by $B^* = (b_1^*, \ldots, b_{\beta(L_0)}^*)$ the solutions provided by FFD and the optimal one in L_0, respectively. For $i = 1, \ldots, \lambda_{\texttt{FFD}}(L_0)$, bins $b_i \in B$ can be seen as subsets of L_0; we suppose that sets b_i are numbered in such a way that: $\forall i \leq \lambda_{\texttt{FFD}}(L_0)$, $\max\{x_k \in L_0 \setminus \cup_{j<i} b_j\} \in b_i$. An analogous numbering is adopted also for sets b_i^*, $1 \leq i \leq \beta(L_0)$ (following this notation, $x_1 \in b_1$ and also $x_1 \in b_1^*$). Then, L_0 satisfies the following.

Lemma 4.

1. $\delta_{\texttt{FFD}}(L_0) \leq 3/4$;
2. *given an optimal solution* $B^* = (b_i^*, i = 1, \ldots, \beta(L_0))$ *of* L_0, *there do not exist two sets of indices* I, J *verifying* $(|I| = |J| \neq 0) \wedge (\cup_{i \in I} b_i = \cup_{j \in J} b_j^*)$;
3. *every optimal solution of* L_0 *is constituted by bins of at least 2 elements; in particular,* $x_1 < x_1 + x_{n_0} \leq 1$;
4. $6 \leq n_0 \leq n$.

Proof. For item 1, it suffices to consider the following instance of size 6: $L = (3/4, 1/2, 1/4 + \epsilon, 1/4 - \epsilon, 1/8, 1/8)$, for $0 \leq \epsilon \leq 1/16$. Then, $\beta(L) = 2$ ($b_1^* = \{3/4, 1/8, 1/8\}$; $b_2^* = \{1/2, 1/4 + \epsilon, 1/4 - \epsilon\}$) and $\lambda_{\texttt{FFD}}(L) = 3$ ($b_1 = \{3/4, 1/4 - \epsilon\}; b_2 = \{1/2, 1/4 + \epsilon, 1/8\}; b_3 = \{1/8\}$). Consequently, $\delta_{\texttt{FFD}}(L) = 3/4$.

For the proof of item 2, let us suppose that such sets of indices exist. Bins b_i^* and b_i being non empty, we have $x = |\cup_{j \in J} b_j^*| \geq 1$; let us then consider $L' = L_0 \setminus \cup_{j \in J} b_j^*$. Of course, L' is not empty: if it was, $\delta_{\texttt{FFD}}(L_0) = 1$, which is not possible (by item 1). But then, by lemma 2, $\delta_{\texttt{FFD}}(L') \leq \delta_{\texttt{FFD}}(L_0)$ and $|L'| < n_0$, a contradiction (L_0 is a minimum-size n-worst instance).

For item 3, let us suppose that L_0 admits an optimal solution $B^* = (b_i^*, i = 1, \ldots, \beta(L_0))$ with a bin $b_l^* = \{x\}$ of cardinality 1. Then, if we denote by b the bin of the FFD-solution which contains x, one can construct an optimal solution $\tilde{B}^*$ admitting b as a bin: $\tilde{B}^* = \{b\} \cup (\cup_{i \neq l} b_i^* \setminus (b \cap b_i^*))$ and contradicting item 2. In the case where $x_1 + x_{n_0} > 1$, x_1 needs a bin for itself in every feasible (in particular in an optimal) solution.

In order to prove item 4, let us consider a 5-worst instance of minimum size denoted by L' (denote by n' the size of L'); of course, the arguments of item 3 also hold for L', in particular, $\beta(L') \leq 2$. Obviously, every instance with optimal value 1

is optimally solved by FFD; hence, without loss of generality, one can suppose that $\beta(L') = 2$.

If $n' = 5$ ($L' = (x_1, \ldots, x_5)$), item 3 implies that every optimal solution of L' consists of a 2-items bin and of a 3-items one. If $b_1^* = \{x_1, x_i\}$, then $\exists j$, $1 < j \leq i$, such that $x_j \in b_1$ but, in this case, the sum of the 3 remaining elements does not exceed the sum of the elements of b_2^*, which implies that $\lambda_{\texttt{FFD}}(L') = 2 = \beta(L')$. Similarly, if $b_1^* = (x_1, x_i, x_j)$, $1 < i < j$, then either $x_i \in b_1$, or $\exists l$, $1 < l < i$, such that $x_l \in b_1$. In the first case, it is obvious that $|b_1| \geq 3$ and that the two other elements have a sum less than 1. In the second case, let k be such that $\{1, \ldots, 5\} = \{1, l, i, j, k\}$; we have then $x_i + x_j + x_k \leq x_i + x_j + x_1 \leq 1$ and, consequently, $\lambda_{\texttt{FFD}}(L') = 2 = \beta(L')$. So, $n' < 5$.

If we suppose $n' = 4$, then item 3 and the fact that $\beta(L') = 2$ imply $b_1^* = \{x_1, x_i\}$ and $b_2^* = \{x_j, x_k\}$; in this case, it follows immediately that $\exists l$, $1 < l \leq i$, such that $x_l \in b_1$ and $\sum_{t \neq 1, t \neq l} x_t \leq 1$, i.e., $\lambda_{\texttt{FFD}}(L') = 2 = \beta(L')$. Consequently, $n' < 4$.

But every instance of size less than or equal to 3 is optimally solved by FFD, this fact implying, first that $n' = 1$ and, second, that every instance of size less than, or equal to, 5 is optimally solved by FFD. This concludes the proof of item 4 and of the lemma. ∎

We are now ready to prove the main result of this paper, expressed by theorem 5.

Theorem 5. *Algorithm FFD achieves differential approximation ratio* 3/4 *in polynomial time. This ratio is tight.*

Proof. Given a size $n \geq 6$, let us consider a n-worst instance L_0 of minimum size n_0. According to items 1 and 4 of lemma 4, $\delta_{\texttt{FFD}}(L_0) \leq 3/4$ and $n_0 \geq 6$.

Let us now prove $\delta_{\texttt{FFD}}(L_0) \geq 3/4$, which implies $\delta_{\texttt{FFD}}(L) \geq 3/4$, $\forall L$. We distinguish different cases according to the structure of the solution $B = (b_1, \ldots, b_{\lambda_{\texttt{FFD}}(L_0)})$.

Case 1: B contains a bin with at least 4 items. Let us denote by b_k such a bin. By the discussion of the previous section, the instance $L' = L_0 \setminus b_k$ satisfies $\beta(L') \leq \beta(L_0)$ and $\lambda_{\texttt{FFD}}(L') = \lambda_{\texttt{FFD}}(L_0) - 1$. Then, a $(0, 1, z)$-application of lemma 3, with $z \geq 4$ gets $\delta_{\texttt{FFD}}(L_0) \geq (z-1)/z \geq 3/4$. This concludes the proof of case 1.

Case 2: B is composed by bins containing at most three items. Here, we distinguish two cases depending on whether $x_1 + x_2 \leq 1$, or $x_1 + x_2 > 1$.

Subcase 2.1: $\mathbf{x_1 + x_2 \leq 1}$. Then, bin b_1 contains x_1 and x_2. Moreover, since the list is sorted in decreasing order, $x_i + x_j \leq 1$, $\forall i \neq j$, which, in particular, implies $|b_2| \geq 2$ (recall that $n_0 \geq 6$).

Sub-subcase 2.1.1: $b_1 \cup b_2$ contains at least five elements. We consider the list $L' = L_0 \setminus (b_1 \cup b_2)$ and prove that

$$\beta(L') \leq \beta(L_0) - 1 \tag{4}$$

Since $x_1 + x_2 \leq 1$ and, moreover, L_0 is sorted in decreasing order, we consider the following three cases: **(i)** $b_1 = \{x_1, x_2, x_3\}$ (i.e., $x_1 + x_2 + x_3 \leq 1$), **(ii)** $x_1 + x_2 + x_3 > 1$ and $x_1 + x_2 + x_{n_0} \leq 1$ ($|b_1| = 3$) and **(iii)** $x_1 + x_2 + x_{n_0} > 1$ and $x_3 + x_4 + x_{n_0} \leq 1$ ($|b_1| = 2, |b_2| = 3$).

*Proof of case **(i)**.* If $b_1 = \{x_1, x_2, x_3\}$, FFD creates bins with three consecutive elements as long as possible and then (recall that $n_0 \geq 6$), $b_2 = \{x_4, x_5, x_6\}$ and

$x_{n_0-6} + \ldots + x_{n_0} > 1$ (if not, the last bin would contain at least four elements); consequently, b_1^* contains at most five elements. Let us now consider the following feasible solution B' for instance L': $B' = (b'_1, \ldots, b'_{\beta(L_0)})$, where $b'_i = b_i^* \setminus (b_i^* \cap \{x_1, \ldots, x_6\})$. If $b'_1 \neq \emptyset$, it is possible to allocate the elements of b'_1 among the positions of elements $x_2, \ldots, x_6$ in B^*. Consequently, $\beta(L') \leq \beta(L_0) - 1$ and this concludes the proof of case **(i)**.

*Proof of case **(ii)**.* In this case, $\{x_1, x_2\} \subset b_1$, $x_3 \in b_2$ and $x_4 \in b_1 \cup b_2$.

Remark 4. Note that FFD is devised in such a way that $b_1 = \{x_1\}$ if and only if $x_1 + x_{n_0} > 1$; consequently, item 3 of lemma 4 implies that b_1 contains at least 2 elements; also by the same item, we have $b_1^* \neq \{x_1\}$. Let us suppose $b_1^* = \{x_1, x_{k_1}, \ldots, x_{k_v}\}$, $k_1 < \ldots < k_v$. ∎

Since $x_1 + x_2 + x_3 > 1 \geq x_1 + x_{k_1} + \ldots + x_{k_v}$, we have $x_2 + x_3 > x_{k_1} + \ldots + x_{k_v}$.

If $k_1 = 2$ (resp., $k_1 = 3$), then elements of $L' \cap \{x_{k_2}, \ldots, x_{k_v}\}$ can be transferred from b_1^* to the position of x_3 (resp., x_2) in B^* in order to constitute a feasible solution of L' using $\beta(L_0) - 1$ bins.

If $k_1 \geq 4$, then let $j = \max\{i \leq v, x_2 \geq x_{k_1} + \ldots + x_{k_i}\}$. If $j = v$ we can, since $k_1 > 2$, transform B^* into a feasible solution of L' using $\beta(L_0) - 1$ bins by transferring elements of $L' \cap \{x_{k_1}, \ldots x_{k_v}\}$ from b_1^* to the position of x_2 in B^*; else, we have $x_2 \geq x_{k_1} + \ldots + x_{k_j}$, $x_3 \geq x_{k_{j+2}} + \ldots + x_{k_v}$ (this sum is eventually null) and, since $k_1 \geq 4$, $x_4 \geq x_{k_{j+1}}$. Once more, it is possible to allocate elements of $L' \cap \{x_{k_1}, \ldots, x_{k_v}\}$ among the positions of $x_1, \ldots, x_4$ in B^*, and this concludes the proof of case **(ii)**.

*Proof of case **(iii)**.* In this case, $b_1 = \{x_1, x_2\}$ and $b_2 = \{x_3, x_4, x_k\}$, $k \geq 5$. By item 2 of lemma 4, $x_2 \notin b_1^*$ (if not, $b_1 = b_1^*$), i.e., $k_1 > 2$. Consequently, $x_1 + x_2 + x_{k_1} > 1 \geq x_1 + x_{k_1} + \ldots + x_{k_v}$, which implies $x_2 > x_{k_2} + \ldots + x_{k_v}$ (this sum is eventually null) and moreover elements $x_{k_2}, \ldots, x_{k_{n_0}}$ can be transferred from b_1^* to the position of x_2 in B^*. Finally, since $x_{k_1} \leq x_3$, if $x_{k_1} \in L'$, we can put it at the place of x_3 in B^* in order to constitute a feasible solution of L' using $\beta(L_0) - 1$ bins, which concludes the proof of case **(iii)** and the proof of expression (4).

Consequently, an $(1, 2, z)$-application of lemma 3 with $z \geq 5$ concludes the proof of sub-subcase 2.1.1.

Sub-subcase 2.1.2: $|b_1 \cup b_2|$ contains at most 4 items. In this case, we have (recall that $n_0 \geq 6$ and that the list is sorted in decreasing order) $b_1 = \{x_1, x_2\}$, $b_2 = \{x_3, x_4\}$ (in particular, $x_1 + x_2 + x_{n_0} \geq x_3 + x_4 + x_{n_0} > 1$) and $\{x_5, x_6\} \subset b_3$. Let us consider the structure of b_1^* and b_2^*. But first we remark the following.

Remark 5. Follow notations for b_1^* adopted in remark 4. Moreover, note that, obviously, $\beta(L_0) \geq 2$ since, in the opposite case, BP would be optimally solved by FFD in L_0. By arguments completely similar to the ones of the proof for item 4 of lemma 4, $|b_2^*| \geq 2$. If x_{l_0} is the greatest element of b_2^*, set $x_{l_0} = x$ and denote $b_2^* = \{x, x_{l_1}, \ldots, x_{l_w}\}$, $l_1 < \ldots < l_w$. ∎

Remark that $x_2 \notin b_1^*$ (i.e., $x_2 \in b_2^*$, $l_0 = 2$) because, in the opposite case, $b_1 = b_1^* = \{x_1, x_2\}$, impossible by item 2 of lemma 4; so, $k_1 \geq 3$. Let us point out that $v \geq 2$ (resp., $w \geq 2$) because, in the opposite case, it would be possible (recall

that $k_1 \geq 3, l_1 \geq 3$) to construct an optimal solution $\tilde{B}^*$ of L_0 with $\tilde{b}_1^* = b_1$ by inverting in B^* elements x_{k_1} and x_2 (resp., x_{l_1} and x_1); this is prohibited by item 2 of lemma 4. Consequently, since $x_1 + x_3 + x_{n_0} \geq x_1 + x_4 + x_{n_0} > 1$ and $x_2 + x_3 + x_{n_0} \geq x_2 + x_4 + x_{n_0} > 1$, we get $k_1 \geq 5$ and $l_1 \geq 5$; hence, setting $r = \min\{k_1, l_1\}$ and $s = \max\{k_1, l_1\}$,

$$x_5 \geq x_r \tag{5}$$

$$x_6 \geq x_s \tag{6}$$

$$x_3 > x_{k_2} + \ldots + x_{k_v} \tag{7}$$

$$x_4 > x_{l_2} + \ldots + x_{l_w} \tag{8}$$

where expression (7) holds thanks to the fact that $x_1 + x_3 + x_{k_1} \geq x_3 + x_4 + x_{n_0} > 1 \geq x_1 + x_{k_1} + \ldots + x_{k_v}$, and expression (8) holds thanks to the fact that $x_2 + x_4 + x_{l_1} \geq x_3 + x_4 + x_n > 1 \geq x_2 + x_{l_1} + \ldots + x_{l_w}$.

Let us now consider the list $L' = L_0 \backslash (b_1 \cup b_2 \cup b_3)$. If $L' \neq \emptyset$, then expressions (5), (6), (7), (8) and the fact that $k_2 \geq 5$ and $l_2 \geq 5$ allow to deduce from B^* a feasible solution of L' using at most $\beta(L_0) - 2$ bins. In both cases: $L' = \emptyset$ and $L' \neq \emptyset$, a $(2, 3, z)$-application of lemma 3 with $z \geq 6$ concludes the proof of the sub-subcase 2.1.2 and of the subcase 2.1.

Subcase 2.2: $\mathbf{x_1 + x_2 > 1}$. This implies $x_2 \in b_2$ and $x_2 \in b_2^*$. We remark that b_2 contains at least 2 elements; indeed, if $b_2 = \{x_2\}$, then (recall that $n_0 \geq 5$ and that every bin of the FFD-solution contains at most three elements) either $b_1 = \{x_1, x_{n_0}\}$ and $x_2 + x_{n_0-1} > 1$, or $b_1 = \{x_1, x_{n_0-1}, x_{n_0}\}$ and $x_2 + x_{n_0-2} > 1$. In both cases, we have necessarily $b_1^* \cup b_2^* \subset b_1 \cup b_2$, which contradicts item 2 of lemma 4.

Let us remark that $x_1 + x_2 > 1$ immediately implies $x_{k_1} + \ldots + x_{k_v} + x_{l_1} + \ldots + x_{l_w} \leq 1$, consequently, denoting $L' = L_0 \setminus \{x_1, x_2\}$, $\beta(L') = \beta(L) - 1$. It immediately follows that, in the case where $|b_1| + |b_2| \geq 5$, we can conclude by a $(1, 2, z)$-application of lemma 3 with $z \geq 5$.

The only remaining possibility to investigate is $|b_1| + |b_2| \leq 4$, i.e., $|b_1| = |b_2| = 2$ (recall that $|b_1| \geq 2$, and $|b_2| \geq 2$). In this case, let us denote $b_1 = \{x_1, x_a\}$ and $b_2 = \{x_2, x_b\}$; clearly, since $n_0 \geq 6$ and since the list is sorted in decreasing order, b_3 contains at least two items; let us also denote by x_{w_1} and by x_{w_2} the two largest elements of b_3.

Unraveling of FFD implies the following: x_{w_1} and x_{w_2} are the two largest elements of the list $L_0 \setminus \{x_1, x_2, x_a, x_b\}$; on the other hand, x_a is the greatest element of L_0 which can be introduced in the same bin as x_1 (in particular, $a \leq k_1$, i.e., $x_a \geq x_{k_1}$), and x_b is the largest element of $L_0 \setminus \{x_a\}$ which can be introduced in the same bin as x_2, i.e., either $a \neq l_1$ and $b \leq l_1$ (then, $x_b \geq x_{l_1}$), or $a = l_1$ and $b \leq a + 1$.

Item 2 of lemma 4 implies that $|b_1^*| \geq 3$ because, in the opposite case, one could invert x_a and x_{k_1} in B^* in order to construct an optimal solution $\tilde{B}^*$ for which $\tilde{b}_1^* = b_1$. Consequently, $v \geq 2$ and $k_v > a$, and the fact that FFD has not introduced x_{k_v} in b_1 means that $x_1 + x_a + x_{k_v} > 1$; so

$$x_a > x_{k_1} + \ldots + x_{k_{v-1}}. \tag{9}$$

If $a \neq l_1$, a very similar argument holds for x_b implying $w \geq 2$ and $b < l_w$. Let us also remark that if $a = l_1$, we also have $w \geq 2$ since, in the opposite case, we

could construct $\tilde{B}^*$ (with $\tilde{b}_1^* = b_1$) by inverting, in B^*, items x_1 and x_2. But then, $b \leq l_1 + 1 \leq l_w$ and, if $b = l_w > a$, $l_w = a + 1 = l_1 + 1$ the fact $|b_2| = 2$ implying in this case (recall that $x_2 + x_{l_1} + x_{l_w} \leq 1$) that $l_w = n_0$ and $a = n_0 - 1$; this contradicts inclusion of x_{k_1} and x_{k_2} in b_1^*. So, always $l_w > b$, therefore

$$x_b > x_{l_1} + \ldots + x_{l_{w-1}}. \tag{10}$$

Considering $L' = L_0 \setminus (b_1 \cup b_2 \cup b_3)$, we are ready to prove $\beta(L') \leq \beta(L_0) - 2$. Indeed, expressions (9) and (10) imply in particular $a \notin \{k_1, \ldots, k_v\}$ and $b \notin \{l_1, \ldots, l_w\}$.

If $\{a, b\} \cap \{k_1, \ldots, k_v, l_1, \ldots, l_w\} = \emptyset$, then expressions (9), (10) and the facts that $x_{w_1} \geq \max\{x_{k_v}, x_{l_w}\}$ and $x_{w_2} \geq \min\{x_{k_v}, x_{l_w}\}$ allow to transform solution B^* into a feasible solution of instance L' using $\beta(L_0) - 2$ bins (this argument holds independently on whether $\{x_{w_1}, x_{w_2}\} \cap (b_1^* \cup b_2^*)$ or not).

If $b = k_i$, $1 \leq i \leq v$, $v \geq 2$, let us choose $i' \in \{1, \ldots, v\}, i' \neq i$. Then, expressions (9) and (10) imply (once more recall that the list is sorted in decreasing order) $x_a > \sum_{j=1}^{w-1} x_{l_j} + \sum_{j \neq i', j \neq i} x_{k_j}$ and $x_{w_1} \geq \max\{x_{l_w}, x_{i'}\}$, $x_{w_2} \geq \min\{x_{l_w}, x_{i'}\}$, so the transformation of B^* described above remains always possible.

For the case $a = l_i$, with arguments completely analogous to the previous ones, we also conclude that $\beta(L') \leq \beta(L_0) - 2$.

A $(2, 3, z)$-application of lemma 3 with $z \geq 6$ concludes the proof of subcase 2.2 ($x_1 + x_2 > 1$) and of case 2. So, $\delta_{\texttt{FFD}}(L_0) \geq 3/4$. Tightness of this ratio follows from the discussion at the beginning of the proof and this completes the proof of theorem 5. ∎

As we have already mentioned, with arguments analogous to the ones of the proof of theorem 5, the same differential approximation bound is achieved by BFD. Furthermore, in order to prove tightness of this bound, we consider list $L = (1/3 + \epsilon, 1/3 + \epsilon, 1/3, 1/3, 1/3 - \epsilon, 1/3 - \epsilon)$. Then, $\beta(L) = 2$ ($b_1^* = b_2^* = \{1/3 - \epsilon, 1/3, 1/3 + \epsilon\}$) and $\lambda_{\texttt{BFD}}(L) = 3$ ($b_1 = \{1/3 + \epsilon, 1/3 + \epsilon\}; b_2 = \{1/3, 1/3, 1/3 - \epsilon\}; b_3 = \{1/3 - \epsilon\}$). Consequently, $\delta_{\texttt{BFD}}(L) = 3/4$ and the following theorem holds.

Theorem 6. $\delta_{\texttt{BFD}} \geq 3/4$ *and it is tight.*

We conclude the section with a result investigating the asymptotic (differential) approximation behavior of FFD. We note that despite of the symmetry between FFD and BFD stated until now, the same result does not hold (at least we have not been able to produce it) for BFD.

Theorem 7. *The asymptotic differential approximation ratio of FFD is bounded below by 7/9.*

Proof. Consider the set $\mathsf{S}_k = \{L : \sigma(L) \geq k\}$, denote by BP($\mathsf{S}_k$) the sub-problem of BP when only lists in S_k are taken into account, and consider a minimal n-worst instance L with respect to (BP(S_k), FFD).

Then, for arbitrarily large values of k, item 3 of lemma 4 applies for BP(S_k); hence, $2\beta(L) \geq \omega(L)$ and since we consider only instances in S_k, the following expressions hold

$$\frac{\beta(L)}{\omega(L) - \beta(L)} \leq 1 \tag{11}$$

$$\omega(L) - \beta(L) \geq k - 1. \tag{12}$$

Moreover, $\forall L$, $\lambda_{\texttt{FFD}}(L)/\beta(L) \leq (11/9) + (4/\beta(L))$ ([5]). Then,

$$\begin{aligned}\delta_{\texttt{FFD}}(L) &= \frac{\omega(L)-\lambda(L)}{\omega(L)-\beta(L)} = 1 - \frac{\lambda(L)-\beta(L)}{\omega(L)-\beta(L)} \\ &\overset{(11)}{\geq} 1 - \frac{2}{9}\frac{\beta(L)}{\omega(L)-\beta(L)} - \frac{4}{\omega(L)-\beta(L)} \overset{(12)}{\geq} \frac{7}{9} - \frac{4}{k-1}.\end{aligned}$$

So finally, $\delta^{\infty}_{\texttt{FFD}} \geq \lim_{k\to\infty} \delta_{\texttt{FFD}}(L) \geq 7/9$, q.e.d. ∎

Theorems 5 and 7 establish an important difference on the behavior of FFD with respect to IFF (and also to FF and BF). In fact, for the latter, δ and δ^{∞} are identical (and equal to 2/3). On the contrary, $\delta_{\text{FFD}} \neq \delta^{\infty}_{\text{FFD}}$. This implies that n-worst instances for (BP, FFD) are of small size, or, more precisely, $\omega(L^w) - \beta(L^w)$ is bounded. In this case, lemma 1 allows to devise polynomial time approximation algorithms for BP guaranteeing differential ratios arbitrarily closed to 7/9. Finally, let us note that, until now, we have not found counter-example proving that 7/9 is tight for FFD.

6 Conclusions

Obviously, n-worst instances have played a key-role in the proofs of the results of this paper. The scope of this notion is not limited to BP and to the differential approximation ratio. It is clear from definition 1 that worst instances of a given size can be defined for every triple $(\Pi, \texttt{A}, \mu)$, where μ can be any approximation measure.

Let us denote standard approximation ratios by ρ and differential approximation ones by δ. It is well-known that $\rho_{\texttt{FF}} = \rho_{\texttt{BF}} = 7/4$, while $\rho_{\texttt{FFD}} = \rho_{\texttt{BFD}} = 3/2$ ([8]), in other words, FF and BF, on the one hand, and FFD and BFD, on the other hand, have identical standard approximation ratios; the same holds, as we have seen above, for the differential approximation framework. Moreover, as it is proved in [6], for $n \geq 6$, the sets of minimal n-worst instances for $(\text{BP}, \texttt{FFD}, \rho)$ and $(\text{BP}, \texttt{FFD}, \delta)$ are identical. The same holds also for BFD. Furthermore, for both approximation measures, the minimal n-worst instances for BFD are equally minimal n-worst ones for FFD, while the converse is not true (it suffices to revisit the list L in the proof of item 1 in lemma 4; applying BFD one gets $\delta_{\texttt{BFD}}(L) = 1$).

In any case, we feel that n-worst instance families are very interesting in polynomial time approximation since they allow to produce positive approximation results by restricting the analysis of an algorithm in the set of worst instances associated with it. As it is shown in [6], the family of n-worst instances can be embedded into a larger one, called "family of critical instances", the importance of which in evaluating or improving the approximation performance of some algorithms is crucial. To our knowledge, no previous research has systematically used notions

closed to the one of n-worst instance in order to formally study the approximation behavior of algorithms and to produce positive approximability results; in this sense, this notion merits further deepening and research. Moreover, if one succeeds to formally characterize the minimal n-worst instances, then the scope of both notions can become even larger, since they could be used even to produce several inapproximability results.

References

[1] G. Ausiello, A. D'Atri, and M. Protasi. Structure preserving reductions among convex optimization problems. *J. Comput. System Sci.*, 21:136–153, 1980.

[2] M. Demange, P. Grisoni, and V. T. Paschos. Differential approximation algorithms for some combinatorial optimization problems. *Theoret. Comput. Sci.*, 209:107–122, 1998.

[3] M. Demange and V. T. Paschos. On an approximation measure founded on the links between optimization and polynomial approximation theory. *Theoret. Comput. Sci.*, 158:117–141, 1996.

[4] M. Demange and V. T. Paschos. Asymptotic differential approximation ratio: definitions, motivations and application to some combinatorial problems. *RAIRO Oper. Res.*, 33:481–507, 1999.

[5] M. R. Garey and D. S. Johnson. *Computers and intractability. A guide to the theory of NP-completeness*. W. H. Freeman, San Francisco, 1979.

[6] J. Monnot. Families of critical instances and polynomial approximation. PhD Thesis. LAMSADE, Université Paris-Dauphine, 1998 (in French).

[7] C. H. Papadimitriou and K. Steiglitz. *Combinatorial optimization: algorithms and complexity*. Prentice Hall, New Jersey, 1981.

[8] D. Simchi-Levi. New worst-case results for the bin-packing problem. *Naval Res. Logistics*, 41:579–585, 1994.