Near-Optimal Algorithms for Stochastic Online Bin Packing

Nikhil Ayyadevara* ✉
Indian Institute of Technology Delhi, India

Rajni Dabas* ✉
Department of Computer Science, University of Delhi, India

Arindam Khan ✉
Department of Computer Science and Automation, Indian Institute of Science, Bengaluru, India

K. V. N. Sreenivas ✉
Department of Computer Science and Automation, Indian Institute of Science, Bengaluru, India

Abstract

We study the online bin packing problem under two stochastic settings. In the bin packing problem, we are given n items with sizes in $(0, 1]$ and the goal is to pack them into the minimum number of unit-sized bins. First, we study bin packing under the i.i.d. model, where item sizes are sampled independently and identically from a distribution in $(0, 1]$. Both the distribution and the total number of items are unknown. The items arrive one by one and their sizes are revealed upon their arrival and they must be packed immediately and irrevocably in bins of size 1. We provide a simple meta-algorithm that takes an offline α-asymptotic proximation algorithm and provides a polynomial-time $(\alpha + \varepsilon)$-competitive algorithm for online bin packing under the i.i.d. model, where $\varepsilon > 0$ is a small constant. Using the AFPTAS for offline bin packing, we thus provide a linear time $(1 + \varepsilon)$-competitive algorithm for online bin packing under i.i.d. model, thus settling the problem.

We then study the random-order model, where an adversary specifies the items, but the order of arrival of items is drawn uniformly at random from the set of all permutations of the items. Kenyon's seminal result [SODA'96] showed that the Best-Fit algorithm has a competitive ratio of at most 3/2 in the random-order model, and conjectured the ratio to be ≈ 1.15. However, it has been a long-standing open problem to break the barrier of 3/2 even for special cases. Recently, Albers et al. [Algorithmica'21] showed an improvement to 5/4 competitive ratio in the special case when all the item sizes are greater than 1/3. For this special case, we settle the analysis by showing that Best-Fit has a competitive ratio of 1. We also make further progress by breaking the barrier of 3/2 for the *3-Partition* problem, a notoriously hard special case of bin packing, where all item sizes lie in $(1/4, 1/2]$.

2012 ACM Subject Classification Theory of computation → Online algorithms

Keywords and phrases Bin Packing, 3-Partition Problem, Online Algorithms, Random Order Arrival, IID model, Best-Fit Algorithm

Digital Object Identifier 10.4230/LIPIcs.ICALP.2022.12

Category Track A: Algorithms, Complexity and Games

Related Version *Full Version*: `http://arxiv.org/abs/2205.03622`

Funding *Arindam Khan*: Research partly supported by Pratiksha Trust Young Investigator Award, Google India Research Award, and Google ExploreCS Award.

Acknowledgements We thank Susanne Albers, Leon Ladewig, and Jirí Sgall for helpful initial discussions. We also thank three anonymous reviewers for their comments and suggestions.

* A part of this work was done when Nikhil and Rajni were summer interns at the Indian Institute of Science, Bengaluru. Rajni's internship was supported by ACM India Anveshan Setu Fellowship.

1 Introduction

Bin Packing (BP) is a fundamental NP-hard combinatorial optimization problem. In BP, we are given a set of items $I := (x_1, x_2, \ldots, x_n)$ with their associated weights (also called sizes) x_i's in $(0, 1]$ and the goal is to partition them into the minimum number of sets (bins) such that the total weight of each set is at most 1. The problem has numerous applications in logistics, scheduling, cutting stock, etc. [10]. Theoretically, bin packing has been the cornerstone for approximation and online algorithms and the study of the problem has led to the development of several interesting techniques [31, 15, 33].

Generally, the performance guarantee of an offline (resp. online) bin packing algorithm $\mathcal{A}$ is measured by asymptotic approximation ratio (AAR) (resp. competitive ratio (CR)). Let $\mathrm{Opt}(I)$ and $\mathcal{A}(I)$ are the objective values returned by the optimal (offline) algorithm and algorithm $\mathcal{A}$, respectively, on input I. Then AAR (resp. CR) is defined as $R^{\infty}_{\mathcal{A}} := \limsup_{m\to\infty}\left(\sup_{I:\mathrm{Opt}(I)=m} \frac{\mathcal{A}(I)}{\mathrm{Opt}(I)}\right)$. Note that $R^{\infty}_{\mathcal{A}}$ focuses on instances where $\mathrm{Opt}(I)$ is large and avoids pathological instances with large approximation ratios where $\mathrm{Opt}(I)$ is small.

Best-Fit (BF), First-Fit (FF), and Next-Fit (NF) are the three most commonly used algorithms for BP. Given x_i as the present item to be packed, they work as follows:

- BF: Pack x_i into the *fullest possible* bin; open a new bin if necessary.
- FF: Pack x_i into the *first possible* bin; open a new bin if necessary.
- NF: Pack x_i into the *most recently opened* bin; open a new bin if necessary.

Johnson et al. [28] studied several heuristics for bin packing such as Best-Fit (BF), First-Fit (FF), Best-Fit-Decreasing (BFD), First-Fit-Decreasing (FFD) and showed their (asymptotic) approximation guarantees to be 17/10, 17/10, 11/9, 11/9, resp. Bekesi et al. [7] gave an $O(n)$ time 5/4-asymptotic approximation algorithm. Another $O(n \log n)$ time algorithm is Modified-First-Fit-Decreasing (MFFD) [29] which attains an AAR of $71/60 \approx 1.1834$. Vega and Lueker [15] gave an asymptotic fully polynomial-time approximation scheme (AFPTAS) for BP: For any $1/2 > \varepsilon > 0$, it returns a solution with at most $(1+\varepsilon)\mathrm{Opt}(I) + O(1)$ [1] bins in time $C_\varepsilon + Cn \log 1/\varepsilon$, where C is an absolute constant and C_ε depends only on ε. Karmarkar and Karp [31] gave an algorithm that returns a solution using $\mathrm{Opt}(I) + O(\log^2 \mathrm{Opt}(I))$ bins. The present best approximation is due to Hoberg and Rothvoss [25] which returns a solution using $\mathrm{Opt}(I) + O(\log \mathrm{Opt}(I))$ bins.

3-Partition problem is a notoriously hard special case of BP where all item sizes are larger than 1/4. Eisenbrand et al. [17] mentioned that "much of the hardness of bin packing seems to appear already in the special case of 3-Partition when all item sizes are in $(1/4, 1/2]$". This problem has deep connections with Beck's conjecture in discrepancy theory [45, 38]. In fact, Rothvoss [25] conjectured that these 3-Partition instances are indeed the hardest instances for bin packing and the additive integrality gap of the bin packing configuration LP for these 3-Parition instances is already $\Theta(\log n)$.

In online BP, items appear one by one and are required to be packed immediately and irrevocably. Lee and Lee [33] presented the Harmonic algorithm with competitive ratio $T_\infty \approx 1.691$, which is optimal for $O(1)$ space algorithms. For general online BP, the present best upper and lower bounds for the CR are 1.57829 [4] and 1.54278 [5], respectively.

In this paper, we focus on online BP under a stochastic setting called the *i.i.d. model* [11] where the input items are sampled from a sequence of independent and identically distributed (i.i.d.) random variables. Here, the performance of an algorithm is measured by the

[1] In bin packing and related problems, the accuracy parameter ε is assumed to be a constant. Here, the term $O(1)$ hides some constants depending on ε.

expected competitive ratio (ECR) $ER_{\mathcal{A}} := \mathbb{E}[\mathcal{A}(I_n(F))]/\mathbb{E}[\text{Opt}(I_n(F))]$, where $I_n(F) := (X_1, X_2, \ldots, X_n)$ is a list of n random variables drawn i.i.d. according to some unknown distribution F with support in $(0, 1]$. Mostly, bin packing has been studied under continuous uniform (denoted by $U[a, b], 0 \leq a < b \leq 1$, where item sizes are chosen uniformly from $[a, b]$) or discrete uniform distributions (denoted by $U\{j, k\}, 1 \leq j \leq k$, where item sizes are chosen uniformly from $\{1/k, 2/k, \ldots, j/k\}$). For $U[0, 1]$, Coffman et al. [13] showed that NF has an ECR of $4/3$ and Lee and Lee [34] showed that the Harmonic algorithm has an ECR of $\pi^2/3 - 2 \approx 1.2899$. Interestingly, Bentley et al. [8] showed that the ECR of FF as well as BF converges to 1 for $U[0, 1]$. It was later shown that the expected wasted space (i.e., the number of needed bins minus the total size of items) is $\Theta(n^{2/3})$ for First-Fit [44, 12] and $\Theta(\sqrt{n}\log^{3/4} n)$ for Best-Fit [44, 35]. Rhee and Talagrand [42] exhibited an algorithm that w.h.p. achieves a packing in $\text{Opt} + O(\sqrt{n}\log^{3/4} n)$ bins for any distribution F on $(0, 1]$. However, note that their competitive ratio can be quite bad when $\text{Opt} \ll n$. A distribution F is said to be *perfectly packable* if the expected wasted space in the optimal solution is $o(n)$ (i.e., nearly all bins in an optimal packing are almost fully packed). Csirik et al. [14] studied the Sum-of-Squares (SS) algorithm and showed that for any perfectly packable distribution, the expected wasted space is $O(\sqrt{n})$. However, for distributions that are not perfectly packable, the SS algorithm has an ECR of at most 3 and can have an ECR of $3/2$ in the worst-case [14]. For any discrete distribution, they gave an algorithm with an ECR of 1 that runs in pseudo-polynomial time in expectation. Gupta et al. [24] also obtained similar $o(n)$ expected wasted space guarantee by using an algorithm inspired by the interior-point (primal-dual) solution of the bin packing LP. However, it remains an open problem to obtain a polynomial-time $(1 + \varepsilon)$-competitive algorithm for online bin packing under the i.i.d. model for arbitrary general distributions. In fact, the present best polynomial-time algorithm for bin packing under the i.i.d. model is BF which has an ECR of at most $3/2$. However, Albers et al. [1] showed that BF has an ECR ≥ 1.1 even for a simple distribution: when each item has size $1/4$ with probability $3/5$ and size $1/3$ with probability $2/5$.

We also study the *random-order model*, where the adversary specifies the items, but the arrival order is permuted uniformly at random. The performance measure in this model is called asymptotic random order ratio (ARR): $RR^{\infty}_{\mathcal{A}} := \limsup_{m\to\infty} \left(\sup_{I:\text{Opt}(I)=m}(\mathbb{E}[\mathcal{A}(I_\sigma)]/\text{Opt}(I))\right)$. Here, σ is drawn uniformly at random from $\mathcal{S}_n$, the set of permutations of n elements, and $I_\sigma := (x_{\sigma(1)}, \ldots, x_{\sigma(n)})$ is the permuted list. Random-order model generalizes the i.i.d. model [1], thus the lower bounds in the random-order model can be obtained from the i.i.d. model. Kenyon in her seminal paper [32] studied Best-Fit under random-order and showed that $1.08 \leq RR^{\infty}_{BF} \leq 3/2$. She conjectured that $RR^{\infty}_{BF} \leq 1.15$. The conjecture, if true, raises the possibility of a better alternate practical offline algorithm: first shuffle the items randomly, then apply Best-Fit. This then beats the AAR of $71/60$ of the present best practical algorithm MFFD. The conjecture has received a lot of attention in the past two decades and yet, no other polynomial-time algorithm is known with a better ARR than BF. Coffman et al. [30] showed that $RR^{\infty}_{NF} = 2$. Fischer and Röglin [21] achieved analogous results for Worst-Fit [27] and Smart-Next-Fit [39]. Recently, Fischer [9] presented an exponential-time algorithm, claiming an ARR of $(1 + \varepsilon)$.

Monotonicity is a natural property of BP algorithms, which holds if the algorithm never uses fewer bins to pack $\hat{I}$ when compared I, where $\hat{I}$ is obtained from I by increasing the item sizes. Murgolo [37] showed that while NF is monotone, BF and FF are not.

Several other problems have been studied under the i.i.d. model and the random-order model [16, 23, 18, 22, 19, 2, 20, 36, 24].

1.1 Our Contributions

Bin packing under the i.i.d. model. We achieve near-optimal performance guarantee for the bin packing problem under the i.i.d. model, thus settling the problem. For any arbitrary unknown distribution F on $(0, 1]$, we give a meta-algorithm (see Algorithm 1) that takes an α-asymptotic approximation algorithm as input and provides a polynomial-time $(\alpha + \varepsilon)$-competitive algorithm. Note that both the distribution F as well as the number of items n are unknown in this case.

▶ **Theorem 1.** *Let $\varepsilon \in (0, 1)$ be a constant parameter. For online bin packing under the i.i.d. model, where n items are sampled from an unknown distribution F, given an offline algorithm $\mathcal{A}_\alpha$ with an AAR of α and runtime $\beta(n)$, there exists a meta-algorithm (Algorithm 1) which returns a solution with an ECR of $(\alpha + \varepsilon)$ and runtime $O(\beta(n))$.* [2]

Using an AFPTAS for bin packing (e.g. [15]) as $\mathcal{A}_\alpha$, we obtain the following corollary.

▶ **Corollary 2.** *Using an AFPTAS for bin packing as $\mathcal{A}_\alpha$ in Theorem 1, we obtain an algorithm for online bin packing under the i.i.d. model with an ECR of $(1 + \varepsilon)$ for any $\varepsilon \in (0, 1/2)$.*

Most algorithms for bin packing under the i.i.d. model are based on the following idea. Consider a sequence of $2k$ items where each item is independently drawn from an unknown distribution F, and let $\mathcal{A}$ be a packing algorithm. Pack the first k items using $\mathcal{A}$; denote the packing by $\mathcal{P}'$. Similarly, let $\mathcal{P}''$ be the packing of the next k items using $\mathcal{A}$. Since each item is drawn independently from F, both $\mathcal{P}'$ and $\mathcal{P}''$ have the same properties in expectation; in particular, the expected number of bins used in $\mathcal{P}'$ and $\mathcal{P}''$ are the same. Thus, intuitively, we want to use the packing $\mathcal{P}'$ as a proxy for the packing $\mathcal{P}''$. However, there are two problems. First, we do not know n, which means that there is no way to know what a good sample size is. Second, we need to show the stronger statement that w.h.p. $\mathcal{P}' \approx \mathcal{P}''$. Note that the items in $\mathcal{P}'$ and $\mathcal{P}''$ are expected to be similar, but they may not be the same. So, it is not clear which item in $\mathcal{P}'$ is to be used as a proxy for a newly arrived item in the second half. Due to the online nature, erroneous choice of proxy items can be quite costly. Different algorithms handle this problem in different ways. Some algorithms exploit the properties of particular distributions, some use exponential or pseudo-polynomial time, etc.

Rhee and Talagrand [41, 42] used *upright matching* to decide which item can be considered as a proxy for a newly arrived item. They consider the model packing $\mathcal{P}_k$ of the first k items (let's call these the proxy items) using an offline algorithm. With the arrival of each of the next k items, they take a proxy item at random and pack it according to the model packing. Then, they try to fit in the real item using upright matching. They repeat this process until the last item is packed. However, they could only show a guarantee of $\mathrm{Opt} + O(\sqrt{n}\log^{3/4} n)$. The main drawback of [42] is that their ECR can be quite bad if $\mathrm{Opt} \ll n$ (say, $\mathrm{Opt} = \sqrt{n}$). One of the reasons for this drawback is that they don't distinguish between small and large items; when there are too many small items, the ECR blows up.

Using a similar approach, Fischer [9] obtained a $(1+\varepsilon)$-competitive randomized algorithm for the random-order model, but it takes exponential time, and the analysis is quite complicated. The exponential time was crucial in finding the optimal packing which was then used as a good proxy packing. However, prior to our work, no polynomial-time algorithm existed which achieves a $(1 + \varepsilon)$ competitive ratio.

[2] As mentioned in an earlier footnote, the $O(\cdot)$ notation hides some constants depending on ε here.

To circumvent these issues, we treat large and small items separately. However, a straightforward adaptation faces several technical obstacles. Thus our analysis required intricate applications of concentration inequalities and sophisticated use of upright matching. First, we consider the semi-random case when we know n. Our algorithm works in stages. For a small constant $\delta \in (0, 1]$, the first stage contains only $\delta^2 n$ items. These items give us an estimate of the distribution. If the packing does not contain too many large items, we show that the simple Next-Fit algorithm suffices for the entire input. Otherwise, we use a proxy packing of the set of first $\delta^2 n$ items to pack the next $\delta^2 n$ items. In the process, we pack the small and large items differently. Then we use the proxy packing of the first $2\delta^2 n$ items to pack the next $2\delta^2 n$ items and so on. In general, we use the proxy packing of the first $2^i\delta^2 n$ items to pack the next $2^i\delta^2 n$ items.

Finally, we get rid of the assumption that we know n using a doubling trick (we guess n first and keep doubling the guess if it turns out to be incorrect). However, there are several difficulties (e.g. input stopping midway of two consecutive guesses, large wasted space in the past stages). Yet, with involved technical adaptations, we can handle them (see Section 2.2).

Our algorithm is simple, polynomial-time (in fact, $O(n)$ time), and achieves essentially the best possible competitive ratio. It is relatively simpler to analyze when compared to Fischer's algorithm [9]. Also, unlike the algorithms of Rhee and Talagrand [42] as well as Fischer [9], our algorithm is deterministic. This is because, unlike their algorithms, instead of taking proxy items at random, we pack all the proxy items before the start of a stage and try to fit in the real items as they come. This makes our algorithm deterministic. Our algorithm is explained in detail in Section 2.1. The nature of the meta-algorithm provides flexibility and ease of application. See Table 1 for the performance guarantees obtained using different offline algorithms.

See Section 2 for the details of the proof and the description of our algorithm. In fact, our algorithm can easily be generalized to d-dimensional online vector packing [6], a multidimensional generalization of bin packing. See the full version for a $d(\alpha+\varepsilon)$ competitive algorithm for d-dimensional online vector packing where the i^{th} item X_i can be seen as a tuple $\left(X_i^{(1)}, X_i^{(2)}, \ldots, X_i^{(d)}\right)$ where each $X_i^{(j)}$ is independently sampled from an unknown distribution $\mathcal{D}^{(j)}$.

Bin packing under the random-order model. Next, we study BP under the random-order model. Recently, Albers et al. [1] showed that BF is monotone if all the item sizes are greater than $1/3$. Using this result, they showed that in this special case, BF has an ARR of at most $5/4$. We show that, somewhat surprisingly, in this case, BF actually has an ARR of 1 (see Section 3.1 for the detailed proof).

▶ **Theorem 3.** *For online bin packing under the random-order model, Best-Fit achieves an asymptotic random-order ratio of* 1 *when all the item sizes are in* $(1/3, 1]$.

Next, we study the 3-partition problem, a special case of bin packing when all the item sizes are in $(1/4, 1/2]$. This is known to be an extremely hard case [25]. Albers et al. [1] mentioned that "it is sufficient to have one item in $(1/4, 1/3]$ to force Best-Fit into anomalous behavior." E.g., BF is non-monotone in the presence of items of size less than $1/3$. Thus the techniques of [1] do not extend to the 3-Partition problem. We break the barrier of $3/2$ in this special case, by showing that BF attains an ARR of ≈ 1.4941.

▶ **Theorem 4.** *For online bin packing under the random-order model, Best-Fit achieves an asymptotic random-order ratio of* ≈ 1.4941 *when all the item sizes are in* $(1/4, 1/2]$.

Table 1 Analysis of Algorithm 1 depending on $\mathcal{A}_\alpha$. In the first row, C is an absolute constant and C_ε is a constant that depends on ε.

$\mathcal{A}_\alpha$	**Time Complexity**	**Expected Competitive Ratio**
AFPTAS [15]	$O(C_\varepsilon + Cn\log 1/\varepsilon)$	$(1+\varepsilon)$
Modified-First-Fit-Decreasing [29]	$O(n\log n)$	$(71/60+\varepsilon)$
Best-Fit-Decreasing [26]	$O(n\log n)$	$(11/9+\varepsilon)$
First-Fit-Decreasing [26]	$O(n\log n)$	$(11/9+\varepsilon)$
Next-Fit-Decreasing [3]	$O(n\log n)$	$(T_\infty+\varepsilon)$
Harmonic [33]	$O(n)$	$(T_\infty+\varepsilon)$
Next-Fit	$O(n)$	$(2+\varepsilon)$

We prove Theorem 4 in Section 3.2. As 3-partition instances are believed to be the hardest instances for bin packing, our result gives a strong indication that the ARR of BF might be strictly less than 3/2.

2 Online Bin Packing Problem under the i.i.d. Model

In this section, we provide the meta algorithm as described in Theorem 1. For the ease of presentation, we split this into two parts. In Section 2.1, we assume a semi-random model, i.e., we assume that the number of items n is known beforehand and design an algorithm. Later, in Section 2.2, we get rid of this assumption.

Let the underlying distribution be F. Without loss of generality, we assume that the support set of F is a subset of $(0,1]$. For any set of items J, we define $\mathcal{W}(J)$ as the sum of weights of all the items in J. For any $k \in \mathbb{N}_+$, we denote the set $\{1,2,\ldots,k\}$ by $[k]$. Let $\varepsilon \in (0,1)$ be a constant parameter and let $0 < \delta < \varepsilon/8$ be a constant. Let $\mathcal{A}_\alpha$ be an offline algorithm for bin packing with an AAR of $\alpha > 1$ and let Opt denote the optimal algorithm. For any $i \in [n]$, we call x_i to be a *large* item if $x_i \geq \delta$ and a *small* item otherwise. Let I_ℓ and I_s denote the set of large and small items in I, respectively.

2.1 Algorithm Assuming that the Value of n is Known

We now describe our algorithm which assumes the knowledge of the number of items. For simplicity, in this section we assume both n and $1/\delta^2$ to be powers of 2. Otherwise, we can round down $\delta \in [1/2^{i+1}, 1/2^i), i \in \mathbb{N}$ to $1/2^{i+1}$. Also, we will anyway get rid of the assumption on the knowledge of n in the next subsection, and there we do not need n to be a power of 2. We denote this algorithm by `Alg`. First, we give a high level idea of the algorithm. We divide the entire input into stages as follows: we partition the input set I into $m := \left(\log(1/\delta^2)+1\right)$ stages $T_0, T_1, \ldots, T_{m-1}$. The zeroth stage T_0, called the *sampling stage*, contains the first $\delta^2 n$ items, i.e., $x_1, x_2, \ldots, x_{\delta^2 n}$. For $j \in [m-1]$, T_j contains the items with index starting from $2^{j-1}\delta^2 n + 1$ till $2^j \delta^2 n$. In essence, T_0 contains the first $\delta^2 n$ items, T_1 contains the next $\delta^2 n$ items, T_2 contains the next $2\delta^2 n$ items, and so on. Note that the number of stages m is a constant. In any stage T_j, we denote the set of large items and small items by L_j and S_j, respectively. For $j \in [m-1]$, let $\overline{T_j}$ denote the set of items which have arrived before the stage T_j, i.e., $\overline{T_j} = T_0 \cup T_1 \cup \cdots \cup T_{j-1}$. Similarly, we define $\overline{L_j}$ and $\overline{S_j}$ as the set of large items and small items, respectively, in $\overline{T_j}$. Note that for any $j \in [m-1]$, $|T_j| = \left|\overline{T_j}\right|$ and since all the items are sampled independently from the same

distribution, we know that with high probability, the optimal solutions of T_j and $\overline{T_j}$ are quite similar. Since $\overline{T_j}$ would have arrived before T_j, we compute an almost optimal packing of $\overline{T_j}$ (in an offline manner) and use it as a blueprint to pack T_j.

The algorithm is as follows: first, we pack T_0, the sampling stage using Next-Fit. The sampling stage contains only a small but a constant fraction of the entire input set; hence it uses only a few number of bins when compared to the final packing but at the same time provides a good estimate of the underlying distribution. If the number of large items in the sampling stage is at most $\delta^3\mathcal{W}(T_0)$, then we continue using Next-Fit for the rest of the entire input too. Intuitively, NF performs well in this case as most of the items are small. Thus, from now on, let us assume otherwise. Now assume that we are at an intermediate point where $\overline{T_j}$ has arrived and T_j is about to arrive ($j \geq 1$). We create D_j, the set of *proxy* items, which is just a copy of $\overline{T_j}$. We pack D_j using $\mathcal{A}_\alpha$. Let this packing be denoted by $\mathcal{P}_j$. Let $B_j^{(k)}$ denote the k^{th} bin in the packing $\mathcal{P}_j$. We iterate over k and remove all the small items in the bin $B_j^{(k)}$ and create a slot in the free space of $B_j^{(k)}$. We call this slot to be an *S-slot*. When an item $x_i \in T_j$ arrives, we check if x_i is small or large

- If x_i is small, we pack it in one of the S-slots greedily, using Next-Fit. If it doesn't fit in any of the S-slots, then we create a new bin with only one S-slot spanning the entire bin (so, this bin will only be used to pack small items), and pack it there.
- If x_i is large, we remove the smallest proxy item with a size more than x_i in the packing $\mathcal{P}_j$ and pack it there. If no such proxy item exists, we open a new bin, pack x_i in there and close it, meaning that it will not be used to pack any further items.

After T_j is packed completely, we just discard the proxy items in the packing that haven't been replaced and move to the next stage. For more formal details and pseudocode for the algorithm, please refer to Algorithm 1.

We will proceed to analyze the algorithm. But first, we will discuss stochastic upright matching and a standard result on it. Using a standard probabilistic concentration lemma, we will formulate a few lemmas which are going to be very important for the analysis of the algorithm.

Stochastic Upright Matching. Rhee and Talagrand [43] studied stochastic upright matching problem in the context of analysis of bin packing algorithms. Consider a set $P = \{(x_i, y_i)\}_{i\in[2n]}$ of $2n$ points where each x_i is either $+1$ or -1 with equal probability and $y_1, y_2, \ldots, y_{2n}$ are sampled according to an i.i.d. distribution. We can define a bipartite graph $\mathcal{G}$ as follows: the vertex set is P, viewed as a set of points in $\mathbb{R}\times\mathbb{R}$. Two points $P_1 = (x_1, y_1), P_2 = (x_2, y_2)$ share an edge iff $x_1 = 1, x_2 = -1$ and $y_1 \geq y_2$.

The objective of the problem is to find a maximum matching in $\mathcal{G}$ or, in other words, minimize the number of unmatched points which we denote by $U(P)$. We denote this matching variant by $\mathcal{M}$. The following lemma shows that w.h.p., the number of unmatched points is $O(\sqrt{n}(\log n)^{3/4})$. The proof of the lemma follows from Lemma 3.1 in [43].

▶ **Lemma 5** ([43]). *Let P be an instance for $\mathcal{M}$. Then there exist constants $a, C, K > 0$ such that, $\mathbb{P}\left[U(P) \geq K\sqrt{n}(\log n)^{3/4}\right] \leq C\exp\left(-a(\log n)^{3/2}\right)$.*

Concentration Inequalities. Now we state the concentration inequalities.

▶ **Lemma 6** (Bernstein's Inequality). *Let $X_1, X_2, \ldots, X_n$ be independent random variables such that each $X_i \in [0,1]$. Then, for any $\lambda > 0$, the following inequality holds.*

$$\mathbb{P}\left[\left|\sum_{i=1}^{n} X_i - \sum_{i=1}^{n}\mathbb{E}\left[X_i\right]\right| \geq \lambda\right] \leq 2\exp\left(-\frac{\lambda^2}{2\left(\sum_{i=1}^{n}\mathbb{E}\left[X_i\right] + \lambda/3\right)}\right).$$

Algorithm 1 $\texttt{Alg}(x_1, x_2, \ldots, x_n)$: A nearly optimal algorithm for online bin packing assuming that the number of items n is known before-hand.

Input: $I_n(\mathcal{D}) = \{x_1, x_2, ..., x_n\}$.
for i= 1 to $\delta^2 n$ **do** ▷ Sampling Stage, T_0
  Pack x_i using NF.
end for
if $|L_0| \leq \delta^3 \mathcal{W}(T_0)$ **then** ▷ Very few large items
  Use NF for all remaining stages.
else
  for j= 1 to $m-1$ **do**
    $D_j \leftarrow \overline{T_j}$; $L(D_j) \leftarrow$ set of large items in D_j.
    Pack D_j using $\mathcal{A}_\alpha$.
    Let the packing be denoted by $\mathcal{P}_j$. ▷ Packing of proxy items
    $\mathcal{S}_j \leftarrow \phi$. ▷ the set of S-slots
    for bin B in $\mathcal{P}_j$ **do**
      Remove the small items in B.
      Create an S-slot H of size equal to (1−weight of all the large items in B).
      $\mathcal{S}_j \leftarrow \mathcal{S}_j \cup H$.
    end for
    for $x_i \in T_j$ **do**
      if x_i is large **then**
        if $\exists d \in L(D_j)$ such that $d \geq x_i$ **then**
          Find smallest such d.
          $L(D_j) \leftarrow L(D_j) \setminus \{d\}$.
          Pack x_i in place of d in the packing $\mathcal{P}_j$.
        else
          Open a new bin and pack x_i and close the bin.
        end if
      else
        Try packing x_i in $\mathcal{S}_j$ using Next-Fit.
        if x_i couldn't be packed **then**
          Open a new bin B' with a single S-slot of unit capacity.
          $\mathcal{S}_j \leftarrow \mathcal{S}_j \cup B'$.
          Pack x_i in B'.
        end if
      end if
    end for
  end for
end if

The following lemma is a direct implication of the results of [42, 40].

▶ **Lemma 7.** *For any $t \in [n]$, let $I(1,t)$ denote the first t items of the input set I. Then there exist constants $K, a > 0$ such that, $\mathbb{P}\left[\mathrm{Opt}(I(1,t)) \geq \frac{t}{n}\mathbb{E}\left[\mathrm{Opt}(I)\right] + K\sqrt{n}(\log n)^{3/4}\right] \leq \exp\left(-a(\log n)^{3/2}\right)$.*

The following lemmas are about how a property of a part of the input (say, the total size) compares to that of the entire input.

▶ **Lemma 8.** *For an input set I of n items drawn from a distribution independently, for any arbitrary set $J \subseteq I$ we have,* $\mathbb{P}\left[\left|\mathcal{W}(J) - \frac{|J|}{n}\mathbb{E}\left[\mathcal{W}(I)\right]\right| \geq \mathbb{E}\left[\mathcal{W}(I)\right]^{2/3}\right] \leq 2\exp\left(-\frac{1}{3}\mathbb{E}\left[\mathcal{W}(I)\right]^{1/3}\right)$.

▶ **Lemma 9.** *Let I be an input set of n items drawn from a distribution independently and let J be any subset of I. Suppose J_ℓ (resp. I_ℓ) denote the set of large items in J (resp. I). Then we have,* $\mathbb{P}\left[\left||J_\ell| - \frac{|J|}{n}\mathbb{E}\left[|I_\ell|\right]\right| \geq \mathbb{E}\left[\mathcal{W}(I)\right]^{2/3}\right] \leq 2\exp\left(-\frac{\delta}{3}\mathbb{E}\left[\mathcal{W}(I)\right]^{1/3}\right)$.

▶ **Lemma 10.** *For any $t \in \{1, 2, \ldots, n\}$, Suppose $I(1,t)$ denote the first t items of the input set I. Then there exist constants $C, a > 0$ such that with probability at least $1 - \exp\left(-a\left(\log \mathbb{E}\left[\mathrm{Opt}(I)\right]\right)^{1/3}\right)$, we have* $\mathrm{Opt}(I(1,t)) \leq (1+2\delta)\frac{t}{n}\mathbb{E}\left[\mathrm{Opt}(I)\right] + C\mathbb{E}\left[\mathrm{Opt}(I)\right]^{2/3}$.

Lemma 7 (resp. Lemmas 8 and 9) intuitively states that the optimal solution for (resp. the weight of the items of, and the number of large items in) a part of the input is almost a linear function of the length of the part. This makes sense because each item comes from the same distribution. Lemma 10 further tries to improve on Lemma 7 by bounding the lower order terms in terms of $\mathbb{E}\left[\mathrm{Opt}(I)\right]$ instead of n while losing only a small factor of $1 + 2\delta$. This is crucial since we need to bound the number of additional bins used by our algorithm in terms of $\mathbb{E}\left[\mathrm{Opt}(I)\right]$ and not in terms of n. We give the proofs of Lemmas 7–10 in the full version. As mentioned, Lemma 7 follows from [42, 40]. Lemmas 8 and 9 are simple applications of Bernstein's inequality. The high-level proof of Lemma 10 goes as follows: first we consider the optimal packing of the large items in $I(1,t)$, then we pack the small items in $I(1,t)$ greedily. If the small items open new bins then it means that all the bins (except one) are filled up to a level at least $1-\delta$. Otherwise, we don't use any extra bins. So, $\mathrm{Opt}(I(1,t)) \leq \max\{\mathrm{Opt}(I_\ell(1,t)), (1+2\delta)\mathcal{W}(I(1,t)) + 1\}$. Then we use Lemmas 7, 8 and 9 to prove Lemma 10.

With these helpful lemmas, we now proceed to analyze the algorithm. We split the analysis into the following two cases: when $|L_0| \leq \delta^3 \cdot \mathcal{W}(T_0)$ and when $|L_0| > \delta^3 \cdot \mathcal{W}(T_0)$.

2.1.1 Case 1: $|L_0| \leq \delta^3 \cdot \mathcal{W}(T_0)$

Recall that in this case, we just continue with Next-Fit for all the remaining items. To bound the Next-Fit solution, we first consider the number of bins that contain at least one large item. For this, we bound the value of $|I_\ell|$. Then we consider the bins that contain only small items and bound this value in terms of weight of all items $\mathcal{W}(I)$.

▷ Claim 11. Let $K := \mathbb{E}\left[\mathrm{Opt}(I)\right]$. For some positive constants C_1, C_2, a, we have that $\mathbb{P}\left[|I_\ell| \leq \delta \cdot \mathcal{W}(T_0) + C_1 K^{2/3}\right] \geq 1 - C_2\exp\left(-aK^{1/3}\right)$.

Proof. As the sampling stage contains $\delta^2 n$ items, $\mathbb{E}\left[|L_0|\right] = \delta^2\mathbb{E}\left[|I_\ell|\right]$. From Lemma 9, we have $\mathbb{P}\left[|L_0| \leq \delta^2\mathbb{E}\left[|I_\ell|\right] - \mathbb{E}\left[\mathcal{W}(I)\right]^{2/3}\right] \leq 2\exp\left(-(\delta/3)\cdot\left(\mathbb{E}\left[\mathcal{W}(I)\right]\right)^{1/3}\right)$, and $\mathbb{P}\left[|I_\ell| \geq \mathbb{E}\left[|I_\ell|\right] + \mathbb{E}\left[\mathcal{W}(I)\right]^{2/3}\right] \leq 2\exp\left(-(\delta/3)\cdot\mathbb{E}\left[\mathcal{W}(I)\right]^{1/3}\right)$. From the above inequalities we have, $|I_\ell| \leq \frac{1}{\delta^2}|L_0| + \left(1+\frac{1}{\delta^2}\right)\mathbb{E}\left[\mathcal{W}(I)\right]^{2/3}$ with probability at least $1 - 4\exp\left(-\frac{\delta}{3}\mathbb{E}\left[\mathcal{W}(I)\right]^{1/3}\right)$. We can use the inequalities $\mathbb{E}\left[\mathrm{Opt}(I)\right] \geq \mathbb{E}\left[\mathcal{W}(I)\right]$ and $|L_0| \leq \delta^3 \cdot \mathcal{W}(T_0)$ to conclude the proof of this claim. ◁

Now we bound the number of bins that are closed by small items. Note that Next-Fit fills each such bin up to a capacity at least $(1-\delta)$. So, the number of such bins is at most $\frac{1}{1-\delta}\mathcal{W}(I)$ when all items are packed by Next-Fit. Also, there can be at most one bin that

can be open. Thus combining all these results, (and using inequality $\frac{1}{1-\delta} \leq 1+2\delta$ for $\delta < \frac{1}{2}$) with high probability, $\mathrm{NF}(I) \leq |I_\ell| + (1+2\delta)\mathcal{W}(I) + 1 \leq \delta \cdot \mathcal{W}(T_0) + (1+2\delta)\mathcal{W}(I) + K\mathbb{E}\left[\mathrm{Opt}(I)\right]^{2/3}$, for some constant K.

Using Lemma 8, we get that with high probability, $\mathcal{W}(I) \leq \mathbb{E}\left[\mathcal{W}(I)\right] + \mathbb{E}\left[\mathcal{W}(I)\right]^{2/3}$. Using the facts $\mathcal{W}(I) \geq \mathcal{W}(T_0)$ and $\mathbb{E}\left[\mathrm{Opt}(I)\right]/2 \leq \mathbb{E}\left[\mathcal{W}(I)\right] \leq \mathbb{E}\left[\mathrm{Opt}(I)\right]$, we get,

$$\mathrm{NF}(I) \leq (1+3\delta)\mathbb{E}\left[\mathrm{Opt}(I)\right] + C_3\mathbb{E}\left[\mathrm{Opt}(I)\right]^{2/3} \tag{1}$$

with probability of at least $1 - C_4 \exp\left(-a_1\mathbb{E}\left[\mathrm{Opt}(I)\right]^{1/3}\right)$ for some constants $C_3, C_4, a_1 > 0$. When the low probability event occurs, we can use the upper bound of $\mathrm{NF}(I) \leq 2\mathrm{Opt}(I) - 1$ to obtain the competitive ratio. Let $p = C_4 \exp\left(-a_1\mathbb{E}\left[\mathrm{Opt}(I)\right]^{1/3}\right)$.

$$\begin{aligned}\mathbb{E}\left[\mathrm{NF}(I)\right] &\leq (1-p)\left((1+3\delta)\mathbb{E}\left[\mathrm{Opt}(I)\right] + K\mathbb{E}\left[\mathrm{Opt}(I)\right]^{2/3} + 1\right) + p(2\mathbb{E}\left[\mathrm{Opt}(I)\right] - 1) \\ &= (1+3\delta+2p)\mathbb{E}\left[\mathrm{Opt}(I)\right] + o(\mathbb{E}\left[\mathrm{Opt}(I)\right])\end{aligned}$$

Since $p = o(1)$ when $\mathbb{E}\left[\mathrm{Opt}(I)\right]$ tends to infinity, we obtain that the expected competitive ratio tends to at most $1 + 3\delta < 1 + \varepsilon$.

2.1.2 Case 2: $|L_0| > \delta^3 \cdot \mathcal{W}(T_0)$

We split our analysis in this case into two parts. We first analyze the number of bins used in the sampling stage T_0 and then analyze the number of bins used in the remaining stages.

Using Lemma 9, we obtain w.h.p. that $|L_0| \leq \delta^2\mathbb{E}\left[|I_\ell|\right] + \mathbb{E}\left[\mathcal{W}(I)\right]^{2/3}$. Hence,

$$\begin{aligned}\mathbb{E}\left[|I_\ell|\right] &\geq \frac{1}{\delta^2}|L_0| - \frac{1}{\delta^2}\mathbb{E}\left[\mathcal{W}(I)\right]^{2/3} \\ &\geq \delta\mathcal{W}(T_0) - \frac{1}{\delta^2}\mathbb{E}\left[\mathcal{W}(I)\right]^{2/3} && \text{(since } |L_0| > \delta^3 \cdot \mathcal{W}(T_0)\text{)} \\ &\geq \delta^3\mathbb{E}\left[\mathcal{W}(I)\right] - \left(\delta + \frac{1}{\delta^2}\right)\mathbb{E}\left[\mathcal{W}(I)\right]^{2/3} && (2)\end{aligned}$$

The last inequality follows from Lemma 8. For any $j \geq 1$, using $|T_j|/n \geq \delta^2$ and using Lemma 9, we get,

$$|L_j| \geq \delta^5\mathbb{E}\left[\mathcal{W}(I)\right] - (2+\delta^3)\mathbb{E}\left[\mathcal{W}(I)\right]^{2/3} \tag{3}$$

Each of the Eqs. (2),(3) holds with a probability of at least $1 - C\exp\left(-a\mathbb{E}\left[\mathcal{W}(I)\right]^{1/3}\right)$ for some constants $C, a > 0$.

Note that $\mathcal{W}(I) \geq \mathrm{Opt}(I)/2$. So from now on we assume that there exist constants $C_1, C_2 > 0$ which depend on δ such that w.h.p. both the following inequalities hold.

$$\mathbb{E}\left[|I_\ell|\right] \geq C_1 \cdot \mathbb{E}\left[\mathrm{Opt}(I)\right] \tag{4}$$

$$|L_j| \geq C_2 \cdot \mathbb{E}\left[\mathrm{Opt}(I)\right] \tag{5}$$

- **Analysis of the Sampling Stage:** Recall that the number of items considered in the sampling phase is $\delta^2 n$. We will bound the number of large items and the weight of items in this stage using Bernstein's inequality.
  1. Since sampling phase has $\delta^2 n$ items, $\mathbb{E}\left[|L_0|\right] = \delta^2\mathbb{E}\left[|I_\ell|\right]$. By applying Bernstein's inequality for $X_1, X_2, \ldots, X_{|T_0|}$ where X_i takes value 1 is x_i is large and 0 otherwise, we get,

$$\begin{aligned}\mathbb{P}\left[|L_0| \geq 2\delta^2\mathbb{E}\left[|I_\ell|\right]\right] &= \mathbb{P}\left[|L_0| \geq \mathbb{E}\left[|L_0|\right] + \delta^2\mathbb{E}\left[|I_\ell|\right]\right] \\ &\leq 2\exp\left(-\frac{\delta^4\mathbb{E}\left[|I_\ell|\right]^2}{2\mathbb{E}\left[|L_0|\right] + \frac{2}{3}\delta^2\mathbb{E}\left[|I_\ell|\right]}\right) \leq 2\exp\left(-\frac{1}{3}\delta^2\mathbb{E}\left[|I_\ell|\right]\right) \\ &\leq 2\exp\left(-a_1 \cdot \mathbb{E}\left[\mathrm{Opt}(I)\right]\right) \qquad \text{(from Equation (4))}\end{aligned}$$

for some constant $a_1 > 0$. So, with high probability, $|L_0| \leq 2\delta^2\mathbb{E}\left[|I_\ell|\right] \leq 2\delta\mathbb{E}\left[\mathrm{Opt}(I)\right]$.

2. Similarly, $\mathbb{E}\left[\mathcal{W}(T_0)\right] = \delta^2\mathbb{E}\left[\mathcal{W}(I)\right]$. By applying Bernstein's inequality for $X_1, X_2, \ldots, X_{|T_0|}$ where X_i takes value x_i, we get,

$$\begin{aligned}\mathbb{P}\left[\mathcal{W}(T_0) \geq 2\delta^2\mathbb{E}\left[\mathcal{W}(I)\right]\right] &= \mathbb{P}\left[\mathcal{W}(T_0) \geq \mathbb{E}\left[\mathcal{W}(S_0)\right] + \delta^2\mathbb{E}\left[\mathcal{W}(I)\right]\right] \\ &\leq 2\exp\left(\frac{-\delta^4\mathbb{E}\left[\mathcal{W}(I)\right]^2}{2\delta^2\mathbb{E}\left[\mathcal{W}(T_0)\right] + \frac{2}{3}\delta^2\mathbb{E}\left[\mathcal{W}(I)\right]}\right) \\ &\leq 2\exp\left(\frac{-\delta^2}{3}\mathbb{E}\left[\mathcal{W}(I)\right]\right) \leq 2\exp\left(\frac{-\delta^2\mathbb{E}\left[\mathrm{Opt}(I)\right]}{6}\right)\end{aligned}$$

So, with high probability we have, $\mathcal{W}(T_0) \leq 2\delta^2\mathbb{E}\left[\mathcal{W}(I)\right] \leq 2\delta^2\mathbb{E}\left[\mathrm{Opt}(I)\right]$.

Since the number of bins opened by Next-Fit $\mathrm{NF}(T_0)$ is at most $|L_0| + \frac{1}{1-\delta}\mathcal{W}(T_0) + 1$, using the bounds on the number of large items and weight of small items in sampling phase, w.h.p. we have,

$$\mathrm{NF}(T_0) \leq 2\delta\mathbb{E}\left[\mathrm{Opt}(I)\right] + \frac{2\delta^2}{1-\delta}\mathbb{E}\left[\mathrm{Opt}(I)\right] \leq 4\delta\mathbb{E}\left[\mathrm{Opt}(I)\right] \tag{6}$$

- **Analysis of the Remaining Stages:** Consider any stage T_j ($j > 0$) after the sampling stage. Note that $\left|\overline{T_j}\right| = |T_j|$. A bin can be opened in three different ways.
  1. When a new bin is opened while packing the set of proxy items D_j using $\mathcal{A}_\alpha$ (see the algorithm description in Section 2.1).
  2. When a large item can't replace a proxy item and hence a new bin is opened for it.
  3. When a small item can't fit in the set of S-slots and hence a new bin is opened with a single S-slot spanning the entire bin.

  In our analysis, first we bound the number of bins opened by $\mathcal{A}_\alpha$ for proxy items; we use Lemma 10 to obtain this bound. Then we show that the number of large items which cannot replace a proxy item would be very small by using upright matching arguments (Lemma 5). For small items, we bound the number of new bins opened by using the fact that $\mathcal{W}(S_j)$ and $\mathcal{W}(\overline{S_j})$ are very close which will be proved using Bernstein's inequality.

  Now we analyze the number of bins opened in the *first way* (say A^j_{proxy}). This is nothing but the number of bins opened by $\mathcal{A}_\alpha$ to pack $\overline{T_j}$. Since, $\mathcal{A}_\alpha$ has an AAR of α, by using Lemma 10, we have,

$$\begin{aligned}A^j_{\mathrm{proxy}} = \mathcal{A}_\alpha(\overline{T_j}) &\leq \alpha\mathrm{Opt}(\overline{T_j}) + o(\mathrm{Opt}(\overline{T_j})) \\ &\leq \alpha(1+2\delta)(\left|\overline{T_j}\right|/n)\mathbb{E}\left[\mathrm{Opt}(I)\right] + C_3\mathbb{E}\left[\mathrm{Opt}(I)\right]^{2/3} + o(\mathrm{Opt}(I)) \qquad (7)\end{aligned}$$

  with probability at least $1 - \exp\left(-a_2\log(\mathbb{E}\left[\mathrm{Opt}(I)\right])^{1/3}\right)$, for some constants $C_3, a_2 > 0$.

  We now bound the number of bins opened in the *second way*. Using Lemma 9, we get that w.h.p., $\left|\overline{L_j}\right| \geq \frac{\left|\overline{T_j}\right|}{n}\mathbb{E}\left[|I_\ell|\right] - \mathbb{E}\left[\mathcal{W}(I)\right]^{2/3}$, and $|L_j| \leq \frac{|T_j|}{n}\mathbb{E}\left[|I_\ell|\right] + \mathbb{E}\left[\mathcal{W}(I)\right]^{2/3}$. Since $\left|\overline{T_j}\right| = |T_j|$ we have,

$$|L_j| \leq \left|\overline{L_j}\right| + 2\mathbb{E}\left[\mathcal{W}(I)\right]^{2/3}. \tag{8}$$

So, w.h.p. the number of large items in T_j doesn't exceed that of those in $\overline{T_j}$ by a large number. Now consider the number of bins opened because there is no feasible proxy item that can be replaced. Let this number be A^j_{unmatch}. We can interpret this number as the number of unmatched items when we use the stochastic matching variant $\mathcal{M}$ from [43] as follows. We can interpret each item $\bar{t} \in \overline{L_j}$ as a point $P_{\bar{t}} := (+1, \bar{t})$ and each point $t \in L_j$ as a point $P_t := (-1, t)$. For simplicity, let's call the points with $+1$ as their first coordinate as *plus* points and the points with -1 as their first coordinate as *minus* points. We match a point $P_{\bar{t}}$ with P_t iff t replaced $\bar{t}$ in our algorithm. It is shown in [44] that such matching is always maximum. Hence the number of items that open new bins is at most the number of unmatched points in this maximum matching. There are two differences though. First, $\left|\overline{L_j}\right|$ may not be greater than $|L_j|$; but as we have shown, w.h.p, the difference can at most be $2\mathbb{E}\left[\mathcal{W}(I)\right]^{2/3}$. Secondly, in the matching variant $\mathcal{M}$, every point has equal chance to be a plus point or minus point. However, this is also inconsequential, since using concentration bounds for binomial random variables, we can show that the number of plus points/minus points lie in the range $\left(\mathbb{E}\left[\left|\overline{L_j}\right|\right] \pm \mathbb{E}\left[\left|\overline{L_j}\right|\right]^{2/3}\right)$ w.h.p. Hence by Lemma 5, we obtain that there exist constants a_3, C_4, K_1 s.t.

$$\mathbb{P}\left[A^j_{\text{unmatch}} \geq K_1\sqrt{\left|\overline{L_j}\right|}\left(\log\left|\overline{L_j}\right|\right)^{3/4} + 2\mathbb{E}\left[\mathcal{W}(I)\right]^{2/3} + 2\mathbb{E}\left[\left|\overline{L_j}\right|\right]^{2/3}\right] \leq C_4\exp\left(-a_3(\log\left|\overline{L_j}\right|)^{3/2}\right)$$

We can simplify the above inequality using Equations (5) and (8) and the fact that $\text{Opt}(I) \leq 2\mathcal{W}(I)$ to obtain that there exist constants $a_4, C_4, K_2 > 0$ such that,

$$\mathbb{P}\left[A^j_{\text{unmatch}} \geq K_2\mathbb{E}\left[\text{Opt}(I)\right]^{2/3}\right] \leq C_4\exp\left(-a_4(\log\mathbb{E}\left[\text{Opt}(I)\right])^{3/2}\right) \tag{9}$$

The only part left is to bound the number of bins opened by small items in *third way*. Let this number be A^j_{small}. We will bound this by using the concentration of weights of small items in $\overline{T_j}$ and T_j. Consider the random variables $X_1, X_2 \ldots X_n$ where $X_i = 0$ if x_i is large, and $X_i = x_i$ otherwise. We have that $\mathcal{W}(S_j) = \sum_{X_i : x_i \in T_j} X_i$ and $\mathcal{W}(\overline{S_j}) = \sum_{X_i : x_i \in \overline{T_j}} X_i$. By applying Bernstein's inequality (similar to Lemma 8) we get, $\mathcal{W}(S_j) \leq \frac{|T_j|}{n}\mathcal{W}(I_s) + \mathbb{E}\left[\mathcal{W}(I)\right]^{2/3}$, and $\mathcal{W}(\overline{S_j}) \geq \frac{|\overline{T_j}|}{n}\mathcal{W}(I_s) - \mathbb{E}\left[\mathcal{W}(I)\right]^{2/3}$ with a probability of at least $1 - C_5\exp\left(-a_5\mathbb{E}\left[\mathcal{W}(I)\right]^{1/3}\right)$ for some constants $C_5, a_5 > 0$. Combining both, we get,

$$\mathcal{W}(S_j) \leq \mathcal{W}(\overline{S_j}) + 2\mathbb{E}\left[\mathcal{W}(I)\right]^{2/3} \tag{10}$$

The initial allocated space for small items at the start of stage j in the packing $\mathcal{P}_j$ is $\mathcal{W}(\overline{S_j})$. Recall that $B^{(k)}_j$ denotes the k^{th} bin in the packing of $\mathcal{P}_j$. While packing the small items, if the S-slot in $B^{(k)}_j$ cannot accommodate a small item, this means that the remaining space in this S-slot is at most δ. So, the weight of small items which overflow the space allocated in packing $\mathcal{P}_j$ is at most $\mathcal{W}(S_j) - \mathcal{W}(\overline{S_j}) + \delta\left|\mathcal{P}_j\right|$ and this entire weight is packed in new bins opened exclusively for small items. Each of these bins (except possibly one) have an occupancy of at least $(1-\delta)$. Since $\mathcal{A}_\alpha$ is α-approximation algorithm, $|\mathcal{P}_j| = \mathcal{A}_\alpha(\overline{T_j}) \leq \alpha\text{Opt}(\overline{T_j}) + o(\text{Opt}(I))$. Using Equations (7) and (10) we get,

$$A^j_{\text{small}} \le \frac{1}{1-\delta}\left(\mathcal{W}(S_j) - \mathcal{W}(\overline{S_j}) + \delta \cdot \alpha \text{Opt}(\overline{T_j}) + o(\text{Opt}(I))\right) + 1$$
$$\le 2\delta \cdot \alpha \frac{|\overline{T_j}|}{n}\mathbb{E}\left[\text{Opt}(I)\right] + C_3\mathbb{E}\left[\text{Opt}(I)\right]^{2/3} + o(\text{Opt}(I)) \tag{11}$$

with high probability. Combining Equations (7), (9), and (11), the number of bins, A_j, opened in the stage j is bounded as,

$$A_j = A^j_{\text{proxy}} + A^j_{\text{unmatch}} + A^j_{\text{small}}$$
$$\le \alpha(1+4\delta)\frac{|\overline{T_j}|}{n}\mathbb{E}\left[\text{Opt}(I)\right] + C_6\mathbb{E}\left[\text{Opt}(I)\right]^{2/3} + o(\text{Opt}(I))$$
$$\le \alpha(1+4\delta)\frac{|T_j|}{n}\mathbb{E}\left[\text{Opt}(I)\right] + C_6\mathbb{E}\left[\text{Opt}(I)\right]^{2/3} + o(\text{Opt}(I)) \tag{12}$$

w.h.p., for some constant C_6. To bound the sum of all A_js, first note that the number of "remaining stages" is $m-1$ which is a constant dependent on δ. Hence, with high probability,

$$\sum_{j=1}^{m-1} A_j \le \alpha(1+4\delta)\sum_{j=1}^{m-1}\frac{|\overline{T_j}|}{n}\mathbb{E}\left[\text{Opt}(I)\right] + (m-1)\cdot C_6\mathbb{E}\left[\text{Opt}(I)\right]^{2/3} + o(\text{Opt}(I))$$
$$\le \alpha(1+4\delta)\mathbb{E}\left[\text{Opt}(I)\right] + C_7\mathbb{E}\left[\text{Opt}(I)\right]^{2/3} + o(\text{Opt}(I)) \tag{13}$$

for some constant $C_7 > 0$ dependent on δ.

In the sampling phase, we have $\text{NF}(T_0) \le 4\delta\mathbb{E}\left[\text{Opt}(I)\right]$ with high probability and in all the remaining phases we have $\sum_{j=1}^{m-1} A_j \le \alpha(1+4\delta)\mathbb{E}\left[\text{Opt}(I)\right] + C_7\mathbb{E}\left[\text{Opt}(I)\right]^{2/3} + o(\text{Opt}(I))$.

Combining both the results we get that w.h.p. the number of bins opened by `Alg` is,

$$\texttt{Alg}(I) \le \alpha(1+8\delta)\mathbb{E}\left[\text{Opt}(I)\right] + C_7\mathbb{E}\left[\text{Opt}(I)\right]^{2/3} + o(\text{Opt}(I))$$
$$\le \alpha(1+\varepsilon)\mathbb{E}\left[\text{Opt}(I)\right] + C_7\mathbb{E}\left[\text{Opt}(I)\right]^{2/3} + o(\text{Opt}(I)) \tag{14}$$

In the low probability event when Equation (14) may not hold, we can bound $\texttt{Alg}(I)$ as follows. In the sampling stage, we have that $\texttt{Alg}(T_0) \le 2\text{Opt}(I) - 1$. For the remaining stages, we bound the number of bins containing at least one large item and the number of bins containing only small items. Because we create a proxy packing at the start of each stage, each large item is packed at most m times. So, the number of bins containing at least one large item is at most $m\,|I_\ell|$. In each stage, with one possible exception, every bin opened which has only small items has an occupancy of at least $(1-\delta)$. Combining over all the stages, the number of bins which contain only small items is at most $\frac{1}{1-\delta}\mathcal{W}(I_s) + m$. Thus, we can bound the total number of bins used by `Alg` to be at most $2\text{Opt}(I) + m\,|I_\ell| + \frac{1}{1-\delta}\mathcal{W}(I_s) + m$. On the other hand, we know that $\text{Opt}(I) \ge \mathcal{W}(I) \ge \delta\,|I_\ell| + \mathcal{W}(I_s)$. Hence, we obtain that $m\,|I_\ell| + \frac{1}{1-\delta}\mathcal{W}(I_s) \le \frac{m}{\delta(1-\delta)}\text{Opt}(I)$. Combining all these, we obtain that

$$\texttt{Alg}(I) \le \left(2 + \frac{m}{\delta(1-\delta)}\right)\text{Opt}(I) + m \tag{15}$$

Now, to obtain the competitive ratio, suppose Equation (14) holds with probability $p(= 1 - o(1))$. We combine Equations (14) and (15) similar to the case when $|L_0| \le \delta^3 \cdot \mathcal{W}(T_0)$.

$$\mathbb{E}\left[\texttt{Alg}(I)\right] \le p(\alpha(1+8\delta)\mathbb{E}\left[\text{Opt}(I)\right] + o(\mathbb{E}\left[\text{Opt}(I)\right]))$$
$$+ (1-p)\left(\left(2 + \frac{m}{\delta(1-\delta)}\right)\mathbb{E}\left[\text{Opt}(I)\right] + m\right)$$
$$\le \alpha(1+\varepsilon)\mathbb{E}\left[\text{Opt}(I)\right] + o(\mathbb{E}\left[\text{Opt}(I)\right]) \qquad (\text{since } 1-p = o(1))$$

Scaling the initial value ε to ε/α before the start of the algorithm, we obtain a competitive ratio of $\alpha + \varepsilon$.

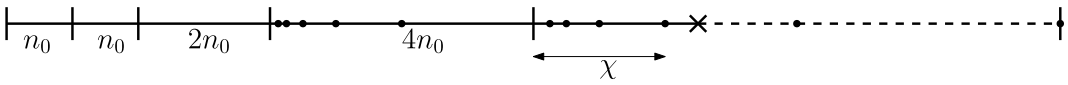

Figure 1 Doubling trick visualized. We start off with an estimate of n to be n_0 and keep doubling it. The first and second super-stages contain n_0 items each. The third super-stage contains $2n_0$ items and so on. Each super-stage is packed individually using `Alg`. The stages in each super-stage are marked by $\bullet$ (for simplicity, only the stages in the last two super-stages are marked). The $\times$ mark indicates the point where the input abruptly stopped. Notice how the fifth stage of the last super-stage is not full. Just before packing this stage, `Alg` constructs a proxy packing of χ by using $\mathcal{A}_\alpha$. But this is very lossy since χ is large compared to this last stage.

2.2 Getting Rid of the Assumption on the Knowledge of the Input Size

In this subsection, we will extend `Alg` for online bin packing with i.i.d. items to devise an algorithm that guarantees essentially the same competitive ratio without knowing the value of n. We denote this algorithm by `ImpAlg`.

We use a doubling trick as follows. We first guess the value of n to be a constant $n_0 := 1/\delta^3$. Then, we run `Alg` until $\min\{n, n_0\}$ items arrive (here, if $\min\{n, n_0\} = n$, then it means that the input stream has ended). If $n > n_0$, i.e., if there are more items to arrive, then we revise our estimate of n by doubling it, i.e., the new value of n is set as $n_1 := 2n_0$. We start `Alg` afresh on the next $\min\{n, n_1\} - n_0$ items. If $n > 2n_0$, then we set the new guess of n to be $n_2 := 2n_1 = 2^2 n_0$ and start `Alg` afresh on the next $\min\{n, n_2\} - n_1$ number of items. We continue this process of doubling our estimate of n until all the items arrive. See Figure 1 for an illustration. The pseudocode is provided in the full version.

We consider the following partition of the entire input into super-stages as follows: The first super-stage, Γ_0, contains the first n_0 items. The second super-stage, Γ_1, contains the next $n_1 - n_0$ items. In general, for $i > 0$, the $(i+1)^{\text{th}}$ super-stage, Γ_i, contains $\min\{n_i, n\} - n_{i-1}$ items which are given by $x_{n_{i-1}+1}, x_{n_{i-1}+2}, \ldots, x_{\min\{n,n_i\}}$. So, essentially, `ImpAlg` can be thought of running `Alg` on each super-stage separately. The number of super-stages is given by $\kappa := \lceil \log(n/n_0) \rceil$. The last super-stage might not be *full*, i.e., ideally, it should contain $n/2$ items but it may not. An even worse case would be when the last stage of the last super-stage is not full, i.e., when it contains fewer than $|\Gamma_{\tau-1}|/2$ items where $\Gamma_{\tau-1}$ is the last super-stage. To see why this is bad, note that since we had assumed the value of n in `Alg`, it was possible to make sure that the last stage had exactly $n/2$ items. This is crucial because our analysis heavily relied on the fact that $|T_j| = \left|\overline{T_j}\right|$ for any stage T_j. This meant that the item set T_j fit almost perfectly in the proxy packing of $\overline{T_j}$. But if the last stage T_{m-1} contains far fewer items than $\overline{T_{m-1}}$, then we might be opening far too many proxy bins than required (see the description of the last super-stage in Figure 1). Hence, we need a slight tweak in `Alg` in the way it packs a stage.

2.2.1 A Tweak to `Alg` in the Way in which a Stage is Packed

Recall that for any $j > 1$, we pack T_j as follows. Just before the stage T_j starts, we pack $\overline{T_j}$ using $\mathcal{A}_\alpha$ (the proxy packing). Instead of packing the entire set $\overline{T_j}$ at once, we do this in chunks. We divide the entire set $\overline{T_j}$ into $\eta := 1/\delta$ number of *equal* chunks. Let's denote these chunks by $\overline{T_j}^{(1)}, \overline{T_j}^{(2)}, \ldots, \overline{T_j}^{(\eta)}$. Suppose each chunk contains c_j number of items. We first pack $\overline{T_j}^{(1)}$ using $\mathcal{A}_\alpha$. Let this packing be denoted by $\mathcal{P}_j^{(1)}$. We pack the first c_j items of T_j by fitting them in the packing $\mathcal{P}_j^{(1)}$ just like we packed a stage in `Alg` by creating S-slots for small items and replacing a proxy item if we need to pack a large item. Then, we pack $\overline{T_j}^{(2)}$

using $\mathcal{A}_\alpha$ and fit the next c_j items of T_j in this packing just like before. We continue this process until all the items in T_j are packed. This way, if the size of T_j is very small compared to that of $\overline{T_j}$, then we only compute the proxy packing of only a few chunks and not of the entire set $\overline{T_j}$. On the other hand, if $|T_j|$ is significant when compared to $\left|\overline{T_j}\right|$, then we are anyway good since the number of chunks η is a constant. Intuitively, the optimal solution for $\overline{T_j}$ and the union of optimal solutions for η chunks $\overline{T_j}^{(1)}, \overline{T_j}^{(2)}, \ldots, \overline{T_j}^{(\eta)}$ are close enough when η is a constant. See the pseudocode of `PackStage` in the full version for details.

2.2.2 Analysis

The analyses of both `ImpAlg` and the tweak to `Alg` can be found in the full version. The intuition to why the tweak to `Alg` doesn't cause much problem has already been provided above. The analysis of `ImpAlg` is a bit complicated though. When n was known we had $O(1)$ stages. However, now we can have $\kappa := \lceil \log(n/n_0) \rceil$ number of super-stages. There can arise two problems:

- We can not analyze each super-stage individually and then sum up the performance guarantees, as we can not use union bound for a super-constant number of events. Moreover, we can't even use the analysis of `Alg` for the first few super-stages since they might only have a constant number of items. So, we consider the $\kappa_1 := \left\lceil \log(\delta^7 n) \right\rceil$ number of the initial super-stages at a time. We show that these initial super-stages contain only a small fraction of the entire input. Each of the final $(\kappa - \kappa_1)$ super-stages can be individually analyzed using the analysis of `Alg`.
- For each super-stage, we can have a constant number of S-bins (bins which contain only small items) with less occupancy. However, since the number of super-stages itself is a super-constant, this can result in a lot of wasted space. For this, we exploit the monotonicity of NF to ensure that we can pack small items from a super-stage into empty slots for small items from the previous stages.

See the full version for the details.

3 Best-Fit under the Random-Order Model

In this section, we will prove Theorems 3 and 4.

3.1 When Item Sizes are Larger than 1/3

First, let us recall upright matching and a related result that we will be using.

Upright Matching Problem. For a positive integer k, let S_k denote the set of all permutations of $[k]$. Consider a set of points P in the two-dimensional coordinate system. Suppose each item is marked as a *plus* point or a *minus* point. Let P^+ denote the set of plus points and let P^- denote the set of minus points. An edge exists between two points p^+, p^- iff $p^+ \in P^+, p^- \in P^-$ and iff p^+ lies above and to the right of p^-, i.e., both the coordinates of p^+ are greater than or equal to the corresponding coordinates of p^-. The objective is to find a maximum matching in this graph or, in other words, minimize the number of unmatched points. We denote the number of unmatched points by $U(P)$.

We will use the following variant of upright matching to prove the final result. Refer to [9] for the proof of the following lemma.

▶ **Lemma 12** ([9]). *Let $k \in \mathbb{N}$ and let $\mathcal{A} = \{a_1, a_2, \ldots, a_{2k}\}$ such that $a_i \geq a_{k+i}$ for all $i \in [k]$. Define a set of* plus *points $P^+ = \{(i, a_i) : i \in [k]\}$ and a set of* minus *points $P^- = \{(i, a_i) : k < i \leq 2k\}$. Suppose we randomly permute the x-coordinates of*

$P^+ \cup P^-$, i.e., for a uniform random permutation $\pi \in S_{2k}$, we redefine P^+ and P^- as $P^+ = \{(\pi(i), a_i) : i \in [k]\}$ and $P^- = \{(\pi(i), a_i) : k < i \leq 2k\}$. Let $P = P^+ \cup P^-$. Then, there exist universal constants $a, C, K > 0$ such that

$$\mathbb{P}\left[U(P) \geq K\sqrt{k}(\log k)^{3/4}\right] \leq C\exp(-a(\log k)^{3/2}) \tag{16}$$

▶ Remark 13. In the above lemma, if we change the definitions of P^+, P^- to be $P^+_{\text{new}} = \{(-\pi(i), a_i) : i \in [k]\}$, $P^-_{\text{new}} = \{(-\pi(i), a_i) : k < i \leq 2k\}$, the guarantee given by Equation (16) doesn't change since the new set $P^+_{\text{new}} \cup P^-_{\text{new}}$ can be constructed by taking a mirror image of the original set $P^+ \cup P^-$ with respect to the y-axis. Since we consider random permutations, the probability of a set and its mirror image is the same.

With the above lemma and remark at hand, we now proceed to prove Theorem 3. Albers et al. [1] showed that the asymptotic random order ratio of the Best-Fit algorithm is at most 1.25 when all the item sizes are more than 1/3. In this section, we improve it further and show that, Best-Fit for this special case under the random-order model is nearly optimal. We first show that the Modified Best-Fit algorithm [11] is nearly optimal and we analyze this using the above variant of stochastic upright matching. The Modified Best-Fit (MBF) algorithm is the same as BF except that it closes a bin if it receives an item of size less than 1/2. Shor[44] showed that MBF *dominates* BF, i.e., for any instance I, $\text{BF}(I) \leq \text{MBF}(I)$. MBF can be easily reduced to upright matching as follows. Given an instance $I = \{x_1, \ldots, x_n\}$, for any item $x_i \in I$, $x_i \in P^-$ if $x_i \leq 1/2$ with x-coordinate as $-i$ and y-coordinate as x_i, and $x_i \in P^+$ if $x_i > 1/2$ with x-coordinate as $-i$ and y-coordinate as $1 - x_i$. So, any item x_s of size $\leq 1/2$ can be matched with an item x_ℓ of size $> 1/2$ if and only if, x_ℓ arrives before x_s and the remaining space in the bin occupied by x_ℓ is more than the size of x_s.

Define an item x_i as a large item (L) if $x_i > 1/2$; otherwise, as a medium item (M) if $x_i \in (1/3, 1/2]$. We define a bin as LM-bin if it contains one large item and one medium item. We use the following lemma which was proved in [1] using the monotonicity property of BF when all item sizes are more than 1/3.

▶ **Lemma 14** ([1]). *Let I be any list that can be packed into $\text{Opt}(I)$ number of LM-bins. If Best-Fit has an AAR of α for I, then it has an AAR of α for any list of items larger than 1/3 as well.*

Consider an input instance which has an optimal packing containing only LM-bins. Consider the number of bins opened by MBF for such instances. Each large item definitely opens a new bin, and a medium item opens a new bin if and only if it can not be placed along with a large item, i.e., it is "unmatched". So, the number of bins opened by MBF equals (number of large items+number of unmatched medium items).

Now, we will prove our result.

▶ **Theorem 15.** *For any list I of items larger than 1/3, the asymptotic random order ratio $RR^{\infty}_{BF} = 1$.*

Proof. From Lemma 14, it is enough to prove the theorem for any list I that can be packed in $\text{Opt}(I)$ LM-bins. So, we can assume that I has k large items and k medium items where $\text{Opt}(I) = k$. Now consider the packing of MBF for a randomly permuted list I_σ. We have, $\text{MBF}(I_\sigma) = (k + \text{number of unmatched medium items})$. Since the optimal packing has all the items matched, we can reduce the following case into the matching variant in Lemma 12: Let ℓ_i, m_i denote the sizes of the large item and the medium item respectively in the i^{th} bin of the

optimal solution. For $i \in [k]$, we let $a_i = 1 - \ell_i$ and $a_{k+i} = m_i$ and let $\mathcal{A} = \{a_1, a_2, \ldots, a_{2k}\}$. Note that the required condition in Lemma 12, i.e., $a_i \geq a_{k+i}$ is satisfied. The arrival order is randomly sampled from S_{2k}. So, we have

$$\mathrm{MBF}(I_\sigma) = k + \frac{U(P)}{2} \leq k + K\sqrt{k}(\log k)^{3/4}$$

with probability of at least $1 - C\exp(-a(\log k)^{3/2})$ for some universal constants $a, C, K > 0$. Since MBF dominates BF we have

$$\mathbb{P}\left[\mathrm{BF}(I_\sigma) \leq k + K\sqrt{k}(\log k)^{3/4}\right] \geq 1 - C\exp(-a(\log k)^{3/2}).$$

In case the high probability event does not occur, we can use the bound of $\mathrm{BF}(I) \leq 1.7\mathrm{Opt}(I) + 2$. Let $p := C\exp(-a(\log k)^{3/2})$. Then

$$\begin{aligned}\mathbb{E}\left[\mathrm{BF}(I_\sigma)\right] &\leq p(1.7\mathbb{E}\left[k\right] + 2) + (1-p)(\mathbb{E}\left[k\right] + K\sqrt{\mathbb{E}\left[k\right]}(\log \mathbb{E}\left[k\right])^{3/4}) \\ &\leq \mathbb{E}\left[\mathrm{Opt}(I)\right] + o(\mathbb{E}\left[\mathrm{Opt}(I)\right]) \qquad \text{(since } p = o(1)\text{)}\end{aligned}$$

So, we get: $RR^{\infty}_{BF} = \limsup_{k\to\infty}\left(\sup_{I:\mathrm{Opt}(I)=k}(\mathbb{E}[\mathrm{BF}(I_\sigma)]/\mathrm{Opt}(I))\right) = 1$. This completes the proof. ◀

3.2 The 3-Partition Problem under Random-Order Model

In this section, we analyze the Best-Fit algorithm under the random-order model given that the item sizes lie in the range $(1/4, 1/2]$, and thus prove Theorem 4. We call an item *small* if its size lies in the range $(1/4, 1/3]$ and *medium* if its size lies in the range $(1/3, 1/2]$. Let I be the input list of items and let $n := |I|$. Recall that given σ, a uniform random permutation of $[n]$, I_σ denotes the list I permuted according to σ. We denote by $\mathrm{Opt}(I_\sigma)$, the number of bins used in the optimal packing of I_σ and by $\mathrm{BF}(I_\sigma)$, the number of bins used by Best-Fit to pack I_σ. Note that $\mathrm{Opt}(I_\sigma) = \mathrm{Opt}(I)$.

If there exists a set of three small items in I_σ such that they arrive as three consecutive items, we call that set to be an *S-triplet*. We call a bin to be a k-bin if it contains exactly k items, for $k \in \{1, 2, 3\}$. We sometimes refer to a bin by mentioning its contents more specifically as follows: An MS-bin is a 2-bin which contains a medium item and a small item. Similarly, an SSS-bin is a 3-bin which contains three small items. Likewise, we can define an M-bin, S-bin, MM-bin, SS-bin, MMS-bin, and MSS-bin.

Since the item sizes lie in $(1/4, 1/2]$, any bin in the optimal packing contains at most three items. For the same reason, in the packing by Best-Fit, every bin (with one possible exception) contains at least two items. This trivially shows that the ECR of Best-Fit is at most $3/2$. To break the barrier of $3/2$, we use the following observations.

- Any 3-bin must contain a small item.
- So, if the optimal solution contains a lot of 3-bins, then it means that the input set contains a lot of small items.

We will prove that if there exist many small items in the input, then with high probability, in a random permutation of the input, there exist many disjoint S-triplets.

▷ Claim 16. Let m be the number of small items in the input set I, and let X_σ denote the maximum number of mutually disjoint S-triplets in I_σ. Suppose $m \geq cn$ where $c = 0.00033$, then the following statements hold true:

1. $\mathbb{E}\left[X_\sigma\right] \geq m^3/(3n^2) \geq c^3 n/3$.
2. $X_\sigma \geq c^3 n/3 - o(n)$ with high probability.

Then, we prove that Best-Fit packs at least one small item from an S-triplet in a 3-bin or in an SS-bin.

▷ Claim 17. Let $\{S_1, S_2, S_3\}$ be an S-triplet in I such that S_3 follows S_2 which in turn follows S_1. Then, in the final packing of Best-Fit of I, at least one of S_1, S_2, S_3 is packed in a 3-bin or in an SS-bin.

But the number of SS-bins in the final packing of Best-Fit can be at most one. So, we obtain that the number of 3-bins in the Best-Fit packing is significant. With these arguments, the proof of Theorem 4 follows. The detailed proofs of the above two claims and the final theorem is given in the full version.

4 Conclusion

We studied online bin packing under two stochastic settings, namely the i.i.d. model, and the random-order model. For the first setting, we devised a meta-algorithm which takes any offline algorithm $\mathcal{A}_\alpha$ with an AAR of α (where α can be any constant ≥ 1), and produces an online algorithm with an ECR of $(\alpha + \varepsilon)$. This shows that online bin packing under the i.i.d. model and offline bin packing are almost equivalent. Using any AFPTAS as $\mathcal{A}_\alpha$ results in an online algorithm with an ECR of $(1 + \varepsilon)$ for any constant $\varepsilon > 0$. An interesting question of theoretical importance is to find whether achieving an ECR of 1 is possible or not. Another related open question is if we can settle online bin packing under the random-order model as well.

Then, we studied the analysis of the well-known Best-Fit algorithm under the random-order model. First, we proved that the ECR of Best-Fit is equal to one if all the item sizes are greater than $1/3$. Then, we improved the analysis of the Best-Fit from 1.5 to ≈ 1.4941, for the special case when the item sizes are in the range $(1/4, 1/2]$. An open question is to further improve this analysis since these instances are conjectured to be the hardest (offline) instances of bin packing. Another interesting problem would be to improve the lower bound on the ECR of Best-Fit in this model (which is currently 1.1 due to [2]).

References

1 Susanne Albers, Arindam Khan, and Leon Ladewig. Best fit bin packing with random order revisited. *Algorithmica*, 83(9):2833–2858, 2021.

2 Susanne Albers, Arindam Khan, and Leon Ladewig. Improved online algorithms for knapsack and GAP in the random order model. *Algorithmica*, 83(6):1750–1785, 2021.

3 Brenda S Baker and Edward G Coffman, Jr. A tight asymptotic bound for next-fit-decreasing bin-packing. *SIAM Journal on Algebraic Discrete Methods*, 2(2):147–152, 1981.

4 János Balogh, József Békési, György Dósa, Leah Epstein, and Asaf Levin. A new and improved algorithm for online bin packing. In *ESA*, volume 112, pages 5:1–5:14, 2018.

5 János Balogh, József Békési, György Dósa, Leah Epstein, and Asaf Levin. A new lower bound for classic online bin packing. In *WAOA*, volume 11926, pages 18–28. Springer, 2019.

6 Nikhil Bansal, Marek Eliás, and Arindam Khan. Improved approximation for vector bin packing. In *SODA*, pages 1561–1579, 2016.

7 Jozsef Bekesi, Gabor Galambos, and Hans Kellerer. A 5/4 linear time bin packing algorithm. *Journal of Computer and System Sciences*, 60(1):145–160, 2000.

8 Jon Louis Bentley, David S Johnson, Frank Thomson Leighton, Catherine C McGeoch, and Lyle A McGeoch. Some unexpected expected behavior results for bin packing. In *STOC*, pages 279–288, 1984.

9 Carsten Oliver Fischer. *New Results on the Probabilistic Analysis of Online Bin Packing and its Variants*. PhD thesis, Rheinische Friedrich-Wilhelms-Universität Bonn, December 2019.

10 Edward G Coffman Jr, János Csirik, Gábor Galambos, Silvano Martello, and Daniele Vigo. Bin packing approximation algorithms: survey and classification. In *Handbook of combinatorial optimization*, pages 455–531. Springer New York, 2013.

11 Edward G Coffman Jr, David S Johnson, George S Lueker, and Peter W Shor. Probabilistic analysis of packing and related partitioning problems. *Statistical Science*, 8(1):40–47, 1993.

12 Edward G Coffman Jr, David S Johnson, Peter W Shor, and Richard R Weber. Bin packing with discrete item sizes, part ii: Tight bounds on first fit. *Random Structures & Algorithms*, 10(1-2):69–101, 1997.

13 Edward G Coffman Jr, Kimming So, Micha Hofri, and AC Yao. A stochastic model of bin-packing. *Information and Control*, 44(2):105–115, 1980.

14 Janos Csirik, David S Johnson, Claire Kenyon, James B Orlin, Peter W Shor, and Richard R Weber. On the sum-of-squares algorithm for bin packing. *Journal of the ACM (JACM)*, 53(1):1–65, 2006.

15 W Fernandez de la Vega and George S Lueker. Bin packing can be solved within 1+epsilon in linear time. *Combinatorica*, 1(4):349–355, 1981.

16 Brian C Dean, Michel X Goemans, and Jan Vondrák. Approximating the stochastic knapsack problem: The benefit of adaptivity. *Mathematics of Operations Research*, 33(4):945–964, 2008.

17 Friedrich Eisenbrand, Dömötör Pálvölgyi, and Thomas Rothvoß. Bin packing via discrepancy of permutations. In *SODA*, pages 476–481, 2011.

18 Jon Feldman, Aranyak Mehta, Vahab Mirrokni, and Shan Muthukrishnan. Online stochastic matching: Beating 1-1/e. In *FOCS*, pages 117–126, 2009.

19 Thomas S Ferguson. Who solved the secretary problem? *Statistical science*, 4(3):282–289, 1989.

20 Carsten Fischer and Heiko Röglin. Probabilistic analysis of the dual next-fit algorithm for bin covering. In *LATIN*, pages 469–482, 2016.

21 Carsten Fischer and Heiko Röglin. Probabilistic analysis of online (class-constrained) bin packing and bin covering. In *LATIN*, volume 10807, pages 461–474. Springer, 2018.

22 Anupam Gupta, Ravishankar Krishnaswamy, and R Ravi. Online and stochastic survivable network design. *SIAM Journal on Computing*, 41(6):1649–1672, 2012.

23 Anupam Gupta, Amit Kumar, Viswanath Nagarajan, and Xiangkun Shen. Stochastic load balancing on unrelated machines. *Mathematics of Operations Research*, 46(1):115–133, 2021.

24 Anupam Gupta and Sahil Singla. Random-order models. In *Beyond the Worst-Case Analysis of Algorithms*, pages 234–258. Cambridge University Press, 2020.

25 Rebecca Hoberg and Thomas Rothvoss. A logarithmic additive integrality gap for bin packing. In *SODA*, pages 2616–2625, 2017.

26 David S Johnson. *Near-optimal bin packing algorithms*. PhD thesis, Massachusetts Institute of Technology, 1973.

27 David S Johnson. Fast algorithms for bin packing. *J. Comput. Syst. Sci.*, 8(3):272–314, 1974.

28 David S Johnson, Alan Demers, Jeffrey D Ullman, Michael R Garey, and Ronald L. Graham. Worst-case performance bounds for simple one-dimensional packing algorithms. *SIAM Journal on computing*, 3(4):299–325, 1974.

29 David S Johnson and Michael R Garey. A 71/60 theorem for bin packing. *J. Complex.*, 1(1):65–106, 1985.

30 Edward G Coffman Jr, János Csirik, Lajos Rónyai, and Ambrus Zsbán. Random-order bin packing. *Discret. Appl. Math.*, 156(14):2810–2816, 2008.

31 Narendra Karmarkar and Richard M Karp. An efficient approximation scheme for the one-dimensional bin-packing problem. In *FOCS*, pages 312–320, 1982.

32 Claire Kenyon. Best-fit bin-packing with random order. In *SODA*, pages 359–364, 1996.

33 Chan C Lee and Der-Tsai Lee. A simple on-line bin-packing algorithm. *J. ACM*, 32(3):562–572, July 1985.

34 Chan C Lee and Der-Tsai Lee. Robust on-line bin packing algorithms. *Technical Report, Northwestern University*, 1987.

35 Tom Leighton and Peter Shor. Tight bounds for minimax grid matching with applications to the average case analysis of algorithms. *Combinatorica*, 9(2):161–187, 1989.

36 Mohammad Mahdian and Qiqi Yan. Online bipartite matching with random arrivals: an approach based on strongly factor-revealing lps. In *STOC*, pages 597–606, 2011.

37 Frank D Murgolo. Anomalous behavior in bin packing algorithms. *Discret. Appl. Math.*, 21(3):229–243, 1988.

38 Alantha Newman, Ofer Neiman, and Aleksandar Nikolov. Beck's three permutations conjecture: A counterexample and some consequences. In *FOCS*, pages 253–262, 2012.

39 Prakash V Ramanan. Average-case analysis of the smart next fit algorithm. *Inf. Process. Lett.*, 31(5):221–225, 1989.

40 W. T. Rhee. Inequalities for bin packing-iii. *Optimization*, 29(4):381–385, 1994. doi:10.1080/02331939408843965.

41 Wansoo T Rhee and Michel Talagrand. Exact bounds for the stochastic upward matching problem. *Transactions of the American Mathematical Society*, 307(1):109–125, 1988.

42 Wansoo T Rhee and Michel Talagrand. On-line bin packing of items of random sizes, ii. *SIAM Journal on Computing*, 22(6):1251–1256, 1993.

43 Wansoo T Rhee and Michel Talagrand. On line bin packing with items of random size. *Mathematics of Operations Research*, 18(2):438–445, 1993.

44 Peter W Shor. The average-case analysis of some on-line algorithms for bin packing. *Combinatorica*, 6(2):179–200, 1986.

45 Joel Spencer. *Ten lectures on the probabilistic method*. SIAM, 1994.