Fakultät für Informatik
Lehrstuhl für Algorithmen und Komplexität

Online Algorithms for Packing Problems in the Random-Order Model

Leon Ladewig

Vollständiger Abdruck der von der Fakultät für Informatik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzende/-r: Prof. Debarghya Ghoshdastidar, Ph.D.

Prüfende der Dissertation:

1. Prof. Dr. Susanne Albers
2. Prof. Dr. Thomas Kesselheim

Die Dissertation wurde am 09.12.2020 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 25.05.2021 angenommen.

Abstract

In the theory of algorithms, the field of online algorithms deals with decision-making under incomplete information. Formally, an *online problem* can be modeled as a sequence of requests where an *online algorithm* must answer each request immediately and irrevocably. Many real-world optimization problems are inherently online, as decisions must be made despite the lack of required information. Therefore, online problems arise in various economic areas as resource allocation, web advertising, and online market design. As a consequence, online algorithms are subject of extensive research in the community of theoretical computer science.

Introduced by Sleator and Tarjan in 1985, the concept of *competitive analysis* became a common approach to measure performance of online algorithms. Here, an online algorithm is compared to an optimal offline algorithm knowing the entire input in advance. In this framework, considering fully adversarial inputs often leads to strong impossibility results. A natural approach beyond worst-case considerations is the *random-order model*, known from the famous secretary problem and formally introduced by Kenyon in 1996. Here, the input for the online algorithm is taken from a worst-case instance, but the order in which the input items are presented to the online algorithm is drawn uniformly at random. This thesis investigates three online packing problems in the random-order model.

The first technical part of the thesis deals with the *k-secretary problem*. In this natural generalization of the well-known secretary problem, k items can be accepted and the goal is to maximize the total profit of the selected items. The problem is not fully understood for small values of $k \geq 2$. We propose a simple algorithm which beats the threshold of $1/e$ in the competitive ratio already for $k = 2$. Moreover, we revisit an algorithm proposed earlier in the literature and show that the present analysis can be improved.

The problem considered in the second part emerges from the secretary problem by introducing knapsack constraints. We study the *online knapsack problem* in the random-order model, also known as *secretary knapsack problem*. Our main contribution is a novel algorithmic approach that combines two strategies tailored for restricted instances. This outperforms previous work in terms of competitive ratio. At the submission time of this thesis, our result is the state-of-the-art algorithm for the knapsack secretary problem.

Finally, the third part of the thesis revisits *online bin packing* in the random-order model. For the Best Fit algorithm, we present several improved bounds. We study a natural special case where at most two items can be assigned to any bin and also give an improved hardness result for the general case.

Acknowledgments

First and foremost, I would like to thank my advisor Prof. Dr. Susanne Albers for her support and advice during my time as a doctoral candidate at Technical University of Munich. I also want to thank Prof. Dr. Klaus Jansen for introducing me into the field of online and approximation algorithms while I was an undergraduate student at Kiel University.

I am highly grateful to Arindam Khan for co-authoring two of my papers as well as for many insightful discussions.

Many thanks to all of my present and former colleagues at the I14 chair, including (but not limited to) Alexander Eckl, Waldo Gálvez, Maximilian Janke, Luisa Peter, Jens Quedenfeld, Harald Räcke, Kevin Schewior, and Richard Stotz. The time we spent together – during and after work – has been a really enjoyable and fun time for me.

A special thanks goes to Maria Kosche for proofreading parts of this thesis.

Finally, I want to express my gratitude to all my family and friends for their everlasting support and believe in me.

Contents

Chapter 1

Introduction

In our everyday life, we permanently need to make decisions under uncertainty. However, it is hard or even impossible to decide optimally given incomplete or faulty information. In particular, the lack of information concerning future events is challenging in most situations.

Suppose you are looking for a new apartment in Munich. As the information in the advertisements is not always reliable (or simply not insightful), you decide to make a list of n apartments and to inspect each of them. After each inspection, you are able to rank the current apartment against all others seen so far but obviously, not against future ones. Your goal is to select the best apartment, which would be an easy task if you could decide after having seen all n apartments. Unfortunately, each landlord insists on your final decision immediately after the inspection. If you dismiss the current apartment, someone else will get it.

In theoretical computer science, the field of online algorithms deals with algorithmic problems where the input is revealed incrementally to the algorithm. This is in contrast to offline problems, where an algorithm can access the entire input any time. An abstract online problem can be stated as follows: The input is a list $I = (x_1, \ldots, x_n)$ which is revealed in n rounds to the online algorithm. In the i-th round, $1 \leq i \leq n$, the item x_i is revealed. The algorithm needs to make an irrevocable decision for x_i immediately on arrival, without knowledge of the upcoming items $x_{i+1}, \ldots, x_n$. Depending on the problem, the length n of the input is either unknown or revealed to the algorithm from the beginning.

Throughout this thesis, we deal with online variants of combinatorial optimization problems. In such problems, the goal is to find a solution that satisfies all imposed combinatorial constraints and whose objective value, a cost or profit associated with the solution, is optimal. However, as finding an optimal solution is typically impossible in online settings, we resort to algorithms that compute approximate solutions. Depending on the problem statement, we obtain either a (profit) maximization problem or a (cost) minimization problem. While computational efficiency is a crucial issue in the design of offline algorithms, in research on online algorithms, the focus lies on approximating an optimal solution with incomplete information. Therefore, it is typically assumed that online algorithms have unlimited computational power. However, we stress that all online algorithms considered in this thesis are efficient, i.e., run in polynomial time.

Competitive analysis Sleator and Tarjan [44] introduced the notion of *competitive analysis* in 1985. In this type of analysis, the solution of an online algorithm is compared to the solution of an optimal offline algorithm. The worst-case ratio over all inputs yields the *competitive ratio* of the online algorithm. Formally, let us denote by $\mathcal{A}(I)$ the objective value of the solution computed by an online algorithm $\mathcal{A}$ on input I. Let $\mathrm{OPT}(I)$ denote the objective value of a solution computed by an optimal offline algorithm. We say that $\mathcal{A}$ is *α-competitive* if for all inputs I,

$$\begin{cases} \mathcal{A}(I) \geq \alpha \cdot \mathrm{OPT}(I) - \beta & \text{for a maximization problem,} \\ \mathcal{A}(I) \leq \alpha \cdot \mathrm{OPT}(I) + \beta & \text{for a minimization problem,} \end{cases}$$

where α is called the *competitive ratio* and $\beta \geq 0$ is a constant. Depending on the specific problem, different restrictions on β may be imposed. If $\beta = 0$, we say that $\mathcal{A}$ is *strictly α-competitive*.

Adversarial model In the above type of analysis, we can think of a game between the online algorithm and a malicious adversary: The adversary designs a worst-case input for the online algorithm, while the online algorithm seeks for maximizing its performance compared to the offline algorithm, given the adversarial input. The consideration of worst-case inputs often leads to strong impossibility results for online problems [24, 39, 45]. In practical settings, however, such inputs seem rather artificial. Hence, the adversarial model seems overly pessimistic in some cases. For this reason, the interest in alternative performance measures and semi-online models has emerged in the research community during the past years. Here, the goal is to smooth the relation between online algorithm and adversary to some degree, which can be reached from two sides: Either, the adversary is given less control over the input, e.g., by randomizing the input (partially). Or, the online algorithm is empowered, e.g., by additional resources, or the ability to revoke decisions in a limited way. The approach studied in this thesis falls into the first category and is formally described in the following.

Random-order model Kenyon [32] introduced the notion of *random-order performance* as alternative performance measure for online algorithms in 1996. Here, the adversary still specifies the set of input items, but has no control over the order in which they are presented to the algorithm. Instead, the order of arrival is a uniformly random permutation. Formally, let σ be a permutation drawn uniformly at random from the set of all permutations of length n. Let I^σ denote the input list I permuted by σ. We say that $\mathcal{A}$ is *α-competitive in the random-order model* if for all inputs I,

$$\begin{cases} \mathrm{E}[\mathcal{A}(I^\sigma)] \geq \alpha \cdot \mathrm{OPT}(I) - \beta & \text{for a maximization problem,} \\ \mathrm{E}[\mathcal{A}(I^\sigma)] \leq \alpha \cdot \mathrm{OPT}(I) + \beta & \text{for a minimization problem,} \end{cases}$$

where the expectation is over the random permutation σ and α, β are defined as before. If $\mathcal{A}$ is a randomized algorithm, the expectation is also over internal random choices. In the following, we use the term *(uniformly) random order* for an input order obtained from a permutation drawn uniformly at random.

Finally, let us revisit the introductory example of finding an apartment. An easy argument shows that the problem does not admit a constant-competitive algorithm in the adversarial model. However, this model seems overly pessimistic; in practice, at least some randomness will occur. Assuming that the n apartments are inspected in random order, the problem admits an elegant algorithm which succeeds with probability $1/e \approx 0.36$. In fact, the problem is a reformulation of the *secretary problem* [22] (see Problem 2.1) – a classic problem from optimal stopping theory and a prime example for random-order analysis.

1.1 Literature survey

In the following, we give an overview of selected results in the random-order model. For more references, we refer to the recent survey by Gupta and Singla [27]. Unless stated otherwise, all competitive ratios mentioned below hold in the random-order model.

Secretary problems The origins of the secretary problem are a bit obscure, but it dates back to the 1960s or even earlier [18]. Lindley [37] and Dynkin [16] showed that a surprisingly simple algorithm succeeds with probability at least $1/e$ and that this is best possible for $n \to \infty$.

More recently, various generalizations of the secretary problem have become subject of intensive research, also motivated by their applications in online auctions [7]. Arguably one of the most natural variants is the *k-secretary problem* (also known as *multiple-choice secretary problem*), where the goal is to maximize the sum of k selected items. Kleinberg [36] showed that this problem admits an algorithm of competitive ratio $1 - 5/\sqrt{k}$ and that the competitive ratio of any algorithm can be bounded from above by $1 - \Omega(\sqrt{1/k})$. Babaioff et al. [6] proposed the OPTIMISTIC algorithm whose competitive ratio is at least $1/e$ for all $k \geq 1$. However, no rigorous analysis of this algorithm is known. Buchbinder et al. [12] analyzed several variants of the secretary problem using linear programming techniques. In the (J, K)-secretary problem, an algorithm can select J items and the objective is to maximize the number of selected top-K items. This problem was revisited later by Chan et al. [13], who devised an optimal 0.4886-competitive algorithm for the case $J = K = 2$. Based on this result, Chan et al. constructed a 0.4920-competitive algorithm for 2-secretary. Despite the progress of [12, 13], the optimal competitive ratio for k-secretary with $k \geq 2$ is still unknown.

The *matroid secretary problem*, introduced by Babaioff et al. [9], covers

various multiple-choice secretary problems and attracted many researchers during the last decade. In this problem, elements of a matroid arrive in random order. At any time, the selected elements must form an independent set in the underlying matroid and the goal is to maximize the combined value of selected elements. The authors of [9] presented an algorithm of competitive ratio $1/\Omega(\log k)$, where k is the rank of the matroid, and asked whether the problem admits a constant-competitive algorithm for general matroids. This question is known as *matroid secretary conjecture*. Since the introduction of the problem, progress has been made in many special cases (see [8] for a recent survey) and also in the general case; the state-of-the-art algorithm [17] has competitive ratio $1/\Omega(\log \log k)$. Still, the matroid secretary conjecture is considered as an open problem of major interest by the community.

Kesselheim et al. [33] studied the secretary problem in a generalized model where items may arrive in non-uniformly random order.

Knapsack problem and generalized packing problems The online knapsack problem under random arrival order was studied first as *knapsack secretary problem* by Babaioff et al. [6]. The authors presented an algorithm of competitive ratio $1/(10e) \approx 1/27$. Significant progress was enabled by recent results on online linear packing programs under random arrival order. Kesselheim et al. [35] showed that this problem admits a competitive ratio of $1 - O(\sqrt{(\log d)/B})$, where d is the maximum number of resources of a single demand, and B is the ratio between the capacity of a resource and the maximum demand for it. As a byproduct, Kesselheim et al. [35] obtained a (1/8.06)-competitive algorithm for the generalized assignment problem (GAP), which includes the knapsack problem with the special case of a single resource. Naori and Raz [42] studied higher-dimensional GAP in the random-order model. The result from [42], published independently and briefly after our submission [3], coincides with our result in the one-dimensional case.

Bin packing and bin covering In a seminal paper on random-order analysis, Kenyon [32] showed that the competitive ratio of the BEST FIT algorithm is at most 1.5 under random order, which is a significantly stronger guarantee than the tight bound of 1.7 under adversarial order [29]. In Kenyon's paper, the upper bound of 1.5 is complemented by a lower bound of 1.08. For the NEXT FIT algorithm, Coffman et al. [30] showed 2-competitiveness under random order, which coincides with the performance in the adversarial model [28]. Fischer and Röglin [21] observed that other natural algorithms fail to perform better under random order as well. In his PhD thesis, Fischer [19] presented a randomized algorithm for online bin packing in the random-order model with competitive ratio $1 + \varepsilon$ for any $\varepsilon > 0$ and exponential running time. Also bin covering, the dual problem of bin packing, has been studied in the random-order model [14, 20, 21].

Scheduling An extensively studied scheduling problem is the makespan minimization on m identical machines. In 1966, Graham [25] proposed a simple online algorithm called LIST, which is $(2 - 1/m)$-competitive under adversarial order. Osborn and Torng [43] showed that LIST is 2-competitive for $m \to \infty$ under random order as well. A 1.8478-competitive deterministic algorithm for makespan minimization was presented recently by Albers and Janke [2], beating the lower bound for deterministic online algorithms under adversarial order. Molinaro [40] studied load balancing under ℓ_p-norms, a generalization of makespan minimization, in the random-order model. Göbel et al. [24] investigated the problem of minimizing the weighted completion times of jobs scheduled on a single machine.

Graph problems A fundamental online problem is bipartite matching with one-sided arrivals. The famous RANKING algorithm by Karp et al. [31] has competitive ratio exactly $1 - 1/e \approx 0.632$ under adversarial order. Mahdian and Yan [38] showed that this barrier can be broken under random order; here, RANKING is 0.696-competitive. Kesselheim et al. [34] studied edge-weighted matching in bipartite graphs with one-sided arrivals and gave an algorithm of competitive ratio $1/e$. Notably, this is optimal, as the problem includes the secretary problem as a special case. Moreover, online edge-coloring for bipartite graphs has been examined in the random-order setting [1, 10, 11].

1.2 Contribution of this thesis

This publication-based thesis includes three papers [5], [3], and [4], (referred to as Papers A, B, and C, respectively) considering different online packing problems in the random-order model. Papers A and B, concerning the k-secretary problem and the online knapsack problem, deal with the problem of choosing a set of items that satisfies a capacity constraint and maximizes the overall profit. While the corresponding offline problems fall into the class of packing problems, in random-order literature, they are usually seen as incremental generalizations of the secretary problem. Both problems are closely related in their nature and applicable techniques. The problem considered in Paper C is technically different: In bin packing, all items must be assigned to unit-sized bins, and the goal is to use as few bins as possible. Below, the contributions of this thesis are described in more detail.

The k-secretary problem (Paper A)

The k-secretary problem is arguably the most immediate and natural generalization of the secretary problem. Here, n items with profits are presented to the algorithm in uniform random order and the algorithm can accept $k \in \mathbb{N}$ items. The goal is to maximize the total profit of the selected items.

We propose a simple threshold-based algorithm SINGLE-REF for the problem. Due to additional parameters, the algorithm is a generalization of the optimal strategy for the secretary problem, hence, it is optimal for $k = 1$. Already for $k = 2$, its competitive ratio is 0.4119, thus significantly greater than $1/e$. We explicitly evaluate the competitive ratio for $1 \leq k \leq 100$. Moreover, we revisit the OPTIMISTIC algorithm proposed by Babaioff et al. [6] and present a tight analysis for the case $k = 2$, revealing that its competitive ratio is 0.4168 in this case. Thus, we confirm that the bound of $1/e$ from [6] is not tight. Our proof reveals an interesting combinatorial property of the algorithm.

Our results particularly address the setting of small $k \geq 2$, a relevant subclass of the problem which is not fully understood in the literature. The algorithm from [36] only works for $k \geq 25$ and beats the barrier of $1/e$ not before $k = 63$. In [6], the authors present two algorithms, but no analyses beating $1/e$ for any k. Finally, although [13] gives a strong 0.4920-competitive algorithm for 2-secretary, deriving algorithms for $k \geq 3$ from the connection to the (J, K)-secretary problem seems overly involved. In contrast, our algorithm SINGLE-REF beats the barrier of $1/e$ already for $k = 2$ while being an easy, threshold-based algorithm. As noted by Buchbinder et al. [12, p. 191], strong algorithms for the k-secretary problem, even restricted to small k, can be helpful in solving related problems. In fact, our results for the knapsack problem (see below) are partially based on insights into this problem.

The knapsack problem (Paper B)

From a broader perspective, the k-secretary problem is a packing problem where each item consumes the same fraction of $1/k$ of the resource's capacity. By introducing arbitrary item sizes, we obtain the *knapsack problem*: Here, the goal is to select a set of items whose total size does not exceed the resource's capacity and whose total profit is maximized.

We develop a (1/6.65)-competitive algorithm for the online knapsack problem in the random-order model, outperforming the previous best algorithm of competitive ratio 1/8.06 [35]. The improvement is based in two pillars: On one side, we develop a novel algorithmic approach where two algorithms, tailored for specific item classes, are performed sequentially. This concept is clearly different from approaches in earlier work [6, 35], where the algorithms decide for one item class and a corresponding strategy by an initial random choice. Again in contrast to previous work [35], we define large items in a way that feasible packings may contain two large items. We call the resulting problem for large items *2-Knapsack* and present a (1/3.08)-competitive algorithm for the problem. The algorithm mostly ignores item sizes and selects items according to the SINGLE-REF algorithm developed for the k-secretary problem. Using our proposed algorithm for 2-Knapsack in combination with an algorithm from [35] within the sequential framework

yields the overall knapsack algorithm.

Finally, we discuss how the sequential approach can be applied to the generalized assignment problem. Here, multiple resources of different capacities are given; the profit and the size of an item depends on the resource. Our approach yields an algorithm of competitive ratio 1/6.99, improving upon the result of [35].

Bin packing (Paper C)

In Paper C, we revisit the BEST FIT algorithm for online bin packing under random arrival order. We present several improved bounds and make the first progress since Kenyon's seminal paper from 1996 [32].

For the case where all items are larger than 1/3, we show that the competitive ratio is at most 1.25, while 1.5 is a lower bound on the competitive ratio in the adversarial setting. For the proof, we identify certain structures which occur in a random permutation with high probability. These structures ensure that BEST FIT maintains parts of the optimal packing. As a technical side contribution, we show that BEST FIT satisfies a monotonicity property if and only if no items smaller than or equal to 1/3 exist.

On the hardness side, we show that for any $k \in \mathbb{N}$, there exists a list I with $\mathrm{OPT}(I) \geq k$ such that the competitive ratio of BEST FIT under random order is larger than 1.10. This tightens the previously best lower bound of 1.08 from [32].

Finally, we initiate the study of strict competitive ratios for the random-order variant of this problem: The previously mentioned bounds include additive constants of the form $\beta = o(\mathrm{OPT}(I))$, corresponding to the notion of asymptotic approximation ratio in offline approximation algorithms. We show that the strict competitive ratio of BEST FIT is at least 1.3 in the random-order model.

1.3 Thesis outline

The remainder of this publication-based thesis is structured as follows. As a warm-up, we consider the secretary problem in Chapter 2. We present a slightly different analysis of the optimal algorithm, demonstrating some techniques used in Papers A and B. In Chapters 3 to 5, we give high-level descriptions of the results and techniques of Papers A to C, respectively. Finally, Chapter 6 comprises some concluding remarks and open problems regarding the results of this thesis.

In Appendices A to C, we provide Papers A to C, respectively. All papers have been accepted at peer-reviewed conferences and are published in the respective conference proceedings [3–5]. Due to space limits, some technical proofs are not included in [3–5]. For full versions, we refer to the corresponding manuscripts on arXiv:

arXiv:2012.00488 [cs.DS]
arXiv:2012.00497 [cs.DS]
arXiv:2012.00511 [cs.DS]

1.4 Basic definitions

Let $\mathbb{N} = \{1, 2, \ldots\}$ denote the set of natural numbers. For subsets of $\mathbb{N}$, we use the symbols $[n] := \{1, \ldots, n\}$ for any $n \in \mathbb{N}$ and $[a..b] := \{a, a+1 \ldots, b\}$ for any $a, b \in \mathbb{N}$ with $a < b$.

As described above, the online problems considered in this thesis fall into the common framework of a strict online setting (immediate and irrevocable decisions) under random arrival order. For future reference, we give a formal definition of this setting below.

Definition 1.1 (Online problem in the random-order model)**.** Let $x_1, \ldots, x_n$ be the items of the problem input and $\sigma\colon [n] \to [n]$ be a permutation drawn uniformly at random. The items are presented sequentially in n rounds to the algorithm, where in round $i \in [n]$, item $x_{\sigma(i)}$ is revealed together with all of its properties. Immediately after arrival of the current item, the online algorithm needs to make an irrevocable decision about this item, without knowledge of items arriving later.

Chapter 2

Warm-up: The Secretary Problem

In 1960, Martin Gardner published the following problem in his column in the *Scientific American*: Suppose an administrator wants to hire a new secretary. She interviews all n applicants sequentially, where each of the $n!$ orders are equally likely. After each interview, the administrator can rank the current candidate against all seen so far and either chooses the current candidate, or rejects it forever. Her goal is to hire the best of all candidates. A formal description of the problem is given below.

Problem 2.1 (Secretary Problem)
A list of n items with ranks $1, \ldots, n$ is presented to the algorithm according to Definition 1.1. The algorithm knows n and can accept one item. The payoff is one if the algorithm accepts the best candidate and zero otherwise.

The first rigorous solutions are attributed to Dynkin [16] and Lindley [37]. Both authors showed that the secretary problem admits a $(1/e)$-competitive algorithm and that this is best possible for $n \to \infty$.

2.1 Optimal algorithm

The well-known optimal algorithm for the secretary problem is of surprising elegance and can be motivated by the following reasoning: First, note that any reasonable algorithm never accepts an item that is not the best so far (as the payoff is zero in these cases, see Problem 2.1). However, it is also risky to pick an item which is best so far in early rounds, as the best item might arrive afterwards.

In fact, the algorithm from [16, 37] follows this approach: Let $t \in [2..n]$ be a parameter specified later. Within the first $t-1$ rounds, the algorithm rejects all items. This time interval is called the *sampling (phase)*; the goal of this phase is to derive a threshold for accepting an item in later rounds. This threshold is set to the best item that arrived in the sampling phase. From round t on, the algorithm accepts the first item that beats the threshold item. A formal description of the discussed algorithm is given in Algorithm 1.

Parameters: $t \in [2..n]$.
In rounds $1, \ldots, t-1$ **do**
  Reject the current item. ▷ `sampling phase`
In rounds $t, \ldots, n$ **do**
  Let a be the rank of the best item in the sampling phase.
  Accept the current item if its rank is better than a.

Algorithm 1: Secretary algorithm

2.2 Analysis

In the following, we show that Algorithm 1 is $(1/e)$-competitive for $n \to \infty$ with $t \approx n/e$. In fact, a refined analysis shows that for any fixed n, the parameter t can be chosen such that the competitive ratio is at least $1/e$ [23]. The text-book proof of Theorem 2.1 is usually quite short; in the proof given below, we construct the random permutation carefully by a sequence of random events and analyze it based on the hypergeometric distribution. This technique is elementary for the analyses in Papers A and B.

Without loss of generality, we identify an item with its rank, i.e., we assume that the items are $1, \ldots, n$, where 1 is the best item.

Theorem 2.1. *Algorithm 1 has competitive ratio* $1/e - 1/n$ *for* $t = \lceil n/e \rceil$.

Proof. According to the definition of Problem 2.1, the payoff is non-zero if and only if the algorithm selects item 1. Therefore, the competitive ratio of Algorithm 1 corresponds to the probability p_1 of accepting the best item. We analyze this probability in the following.

Let $pos(i)$ denote the position of item i in a (partially) fixed permutation. The following process creates a random permutation with uniform distribution:

1. Draw $pos(1)$ uniformly at random from $[n]$.
2. Draw $pos(a)$ uniformly at random from $[n] \setminus \{pos(1)\}$.
3. Process the remaining free positions in increasing order; for each position, draw an item from $[n] \setminus \{1, a\}$ uniformly at random without replacement.

Let $E_{i,a}$ be the event that item 1 is accepted in round i (with $t \leq i \leq n$) and the best sampling item is $a \geq 2$. With respect to the random process defined above, $E_{i,a}$ is equivalent to the intersection of the following three events:

- $E_{i,a}^{(1)}$: The outcome of step 1 is i.
- $E_{i,a}^{(2)}$: The outcome of step 2 is in $[t-1]$.

- $E_{i,a}^{(3)}$: The items drawn in step 3 before position i have rank larger than a.

The event $E_{i,a}^{(1)}$ happens with probability $1/n$. Similarly, $\Pr[E_{i,a}^{(2)} \mid E_{i,a}^{(1)}] = \frac{t-1}{n-1}$. For $E_{i,a}^{(3)}$ conditioned on $E_{i,a}^{(1)} \cap E_{i,a}^{(2)}$, we investigate the probability of drawing $i-2$ blue balls without replacement from an urn containing exactly $n-a$ blue balls and $n-2$ balls in total. As the number of blue balls in this random sample follows the hypergeometric distribution, the probability is $\binom{n-a}{i-2}/\binom{n-2}{i-2}$. Therefore,

$$\Pr[E_{i,a}] = \frac{1}{n} \cdot \frac{t-1}{n-1} \cdot \frac{\binom{n-a}{i-2}}{\binom{n-2}{i-2}} .$$

Summing over all positions $i \in [t..n]$ and ranks $a \in [2..n]$, we obtain

$$\begin{aligned} p_1 &= \sum_{i=t}^{n} \sum_{a=2}^{n} \Pr[E_{i,a}] \\ &= \sum_{i=t}^{n} \sum_{a=2}^{n} \left(\frac{1}{n} \cdot \frac{t-1}{n-1} \cdot \frac{\binom{n-a}{i-2}}{\binom{n-2}{i-2}} \right) \\ &= \frac{1}{n} \cdot \frac{t-1}{n-1} \cdot \sum_{i=t}^{n} \left(\frac{\sum_{a=2}^{n} \binom{n-a}{i-2}}{\binom{n-2}{i-2}} \right) . \end{aligned}$$

The last term can be simplified by observing

$$\frac{\sum_{a=2}^{n} \binom{n-a}{i-2}}{\binom{n-2}{i-2}} = \frac{\sum_{a=0}^{n-2} \binom{a}{i-2}}{\binom{n-2}{i-2}} = \frac{\binom{n-1}{i-1}}{\binom{n-2}{i-2}} = \frac{n-1}{i-1} ,$$

which gives

$$p_1 = \frac{t-1}{n} \cdot \sum_{i=t}^{n} \frac{1}{i-1} . \tag{2.1}$$

Since $1/(i-1)$ decreases monotonically in i, the sum $\sum_{i=t}^{n} \frac{1}{i-1}$ is bounded from below by the integral $\int_{t}^{n+1} \frac{1}{i-1} \, \mathrm{d}i \; = \ln \frac{n}{t-1}$, see Figure 2.1. Therefore, with $t = \lceil cn \rceil$ for $c \in (0,1)$, we obtain

$$p_1 \geq \frac{\lceil cn \rceil - 1}{n} \cdot \ln \frac{n}{\lceil cn \rceil - 1} \geq \frac{cn-1}{n} \cdot \ln \frac{n}{cn} = c \cdot \ln \frac{1}{c} - \frac{\ln(1/c)}{n} . \tag{2.2}$$

The term $c \cdot \ln(1/c)$ is maximized for $c = 1/e$, see Figure 2.2. Thus, we obtain a competitive ratio of $p_1 \geq 1/e - 1/n$. □

In fact, an equivalent proof can be obtained from a simpler argument: The probability that the best item is accepted in round $i \in [t..n]$ is the

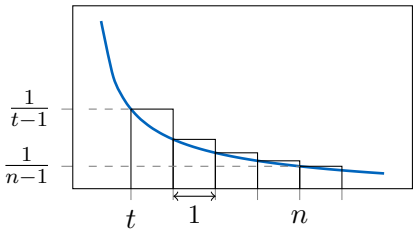


Figure 2.1: Illustration of the bound $\sum_{i=t}^{n} \frac{1}{i-1} \geq \int_{t}^{n+1} \frac{1}{i-1}\,\mathrm{d}i$.

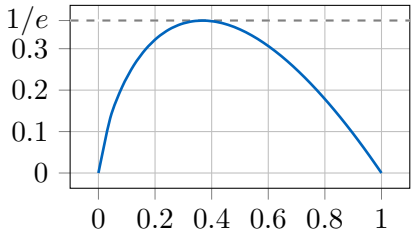


Figure 2.2: Probability p_1 as function of c.

probability that it arrives in this round and no item has been accepted previously, i.e., the best element in rounds $1, \ldots, t-1$ is the best element in rounds $1, \ldots, i-1$. The probability for this event is $\frac{1}{n} \cdot \frac{t-1}{i-1}$. Summing over all positions $i \in [t..n]$ yields Equation (2.1).

Dynkin and Lindley [16, 37] showed that $1/e$ is the best possible competitive ratio for $n \to \infty$. Hence, Algorithm 1 with $t \approx n/e$ is asymptotically optimal. An alternative proof of optimality can be obtained from the linear programming framework for secretary problems by Buchbinder et al. [12].

Chapter 3

The k-secretary Problem

In Paper A, we investigate two algorithms for the k-secretary problem. This chapter provides a high-level description of the approaches and results. We first give a formal problem definition and a few technical preliminaries.

Problem 3.1 (k-secretary Problem)
Let $k \in \mathbb{N}$. A list of n items $1, \ldots, n$ with profits $v_1, \ldots, v_n$ is presented to the algorithm according to Definition 1.1. The algorithm knows n and can accept up to k items. The goal is to maximize the total profit of the accepted items.

We emphasize that the definition of Problem 3.1 allows algorithms to observe the actual profits, unlike Problem 2.1. This corresponds to the problem definition in [36]; however, some papers in the literature require k-secretary algorithms to work only with ordinal information in the following sense.

Definition 3.1. We say that a k-secretary algorithm is *ordinal* if it only uses the relative ranks of the items and not their numerical profits.

Chan et al. [13] showed that ordinal algorithms for the k-secretary problem cannot achieve optimal competitive ratios.

Another natural property of most k-secretary algorithms is *monotonicity*, meaning that the probabilities of selecting items grow monotonically with their profits.

Definition 3.2. We say that a k-secretary algorithm $\mathcal{A}$ is *monotone* if for any two items i and j with profits $v_i \geq v_j$, it holds that $\Pr[\mathcal{A} \text{ accepts } i] \geq \Pr[\mathcal{A} \text{ accepts } j]$.

Next, we introduce some notation used in the following proofs.

Notation Without loss of generality, we assume $v_1 > v_2 > \ldots > v_n$. For a fixed ordinal algorithm $\mathcal{A}$, we define $p_i = \Pr[\mathcal{A} \text{ accepts item } i]$. Let $\mathrm{E}[\mathcal{A}] = \sum_{i=1}^{n} p_i v_i$ denote the expected profit of the solution returned by $\mathcal{A}$, and let $\mathrm{OPT} = \sum_{i=1}^{k} v_i$.

As both algorithms discussed in Sections 3.1 and 3.2 are ordinal and monotone according to Definitions 3.1 and 3.2, we will use the following lemma to analyze their competitive ratios.

Lemma 3.1. *Let $\mathcal{A}$ be an ordinal and monotone algorithm for the k-secretary problem. The competitive ratio of $\mathcal{A}$ is $(1/k) \cdot \sum_{i=1}^{k} p_i$.*

Proof. As $p_1, \ldots, p_k$ and $v_1, \ldots, v_k$ are non-increasing sequences,

$$\frac{1}{k} \cdot \sum_{i=1}^{k} p_i v_i \geq \left(\frac{1}{k} \cdot \sum_{i=1}^{k} p_i \right) \cdot \left(\frac{1}{k} \cdot \sum_{i=1}^{k} v_i \right)$$

holds by Chebyshev's sum inequality [26]. Applying this inequality yields

$$\mathrm{E}[\mathcal{A}] = \sum_{i=1}^{n} p_i v_i \geq \sum_{i=1}^{k} p_i v_i \geq \frac{1}{k} \cdot \left(\sum_{i=1}^{k} p_i \right) \cdot \left(\sum_{i=1}^{k} v_i \right) = \left(\frac{1}{k} \cdot \sum_{i=1}^{k} p_i \right) \cdot \mathrm{OPT}\,.$$

Thus, the competitive ratio of $\mathcal{A}$ is at least $(1/k) \cdot \sum_{i=1}^{k} p_i$. Now, consider a collection of items with $v_i \approx 1$ for $1 \leq i \leq k$ and $v_i \approx 0$ for $k+1 \leq i \leq n$. Here, all inequalities used above are tight. □

3.1 The Single-Ref algorithm

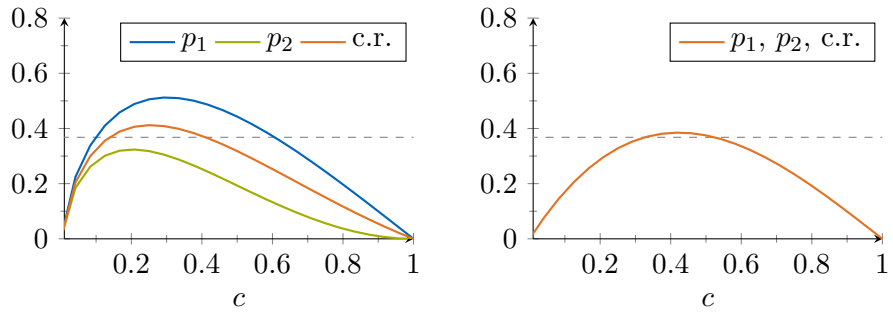
A natural algorithm for the k-secretary problem can be obtained from the optimal strategy for the secretary problem discussed in Chapter 2: After an initial sampling phase of $t-1$ rounds, with $2 \leq t \leq n$, a threshold item v^+ is determined and the first k items beating v^+ are accepted subsequently. This is the underlying idea of the SINGLE-REF algorithm developed in Paper A. Here, the threshold item is the r-th best item from the sampling phase, where $r \in [k]$ is a parameter of the algorithm. Finally, r will be chosen depending on k in order to derive a reasonable threshold. A formal description of SINGLE-REF is given in Algorithm 2.

Parameters: $k \in \mathbb{N}$, $r \in [k]$, $t \in [2..n]$.
In rounds $1, \ldots, t-1$ **do**
  Reject the current item. ▷ `sampling phase`
In rounds $t, \ldots, n$ **do**
  Let s_r be the r-th best profit from the sampling phase.
  Accept the current item if its profit exceeds s_r (and at most $k-1$ items have been accepted).

Algorithm 2: The SINGLE-REF algorithm

We call Algorithm 2 the SINGLE-REF algorithm, as a single reference element serves as the threshold for accepting items. This is in contrast to other approaches [6, 13, 36], where several items are involved. Note that Algorithm 2 includes Algorithm 1 with the special case $k = r = 1$, thus, for $t \approx n/e$ and $n \to \infty$, Algorithm 2 is optimal for the secretary problem.

In the following, we assume $t - 1 = cn$ for $c \in (0, 1)$, i.e., we assume that the length of the sampling phase is a constant fraction c of the input length.



(a) $r = 1$ **(b)** $r = 2$

Figure 3.1: Probabilities p_1, p_2, and competitive ratio (c.r.) of SINGLE-REF as functions of c in the case $k = 2$. The gray dashed line represents the value $1/e$.

3.1.1 Analysis

The analysis of SINGLE-REF is based on the observation that the parameter r partitions the items from $[k]$ into two classes $[1..r]$ and $[r+1..k]$: Any item $i \in [r]$ belongs to the class of the r (globally) best items, hence, $v_i \geq s_r$. Therefore, i dominates the reference element (unless $v_i = s_r$); it is accepted whenever it occurs after the sampling and at most $k-1$ items have been accepted previously. We call the items $1, \ldots, r$ *dominating items*. As the algorithm does not distinguish between any two dominating items, we have $p_i = p_1$ for any dominating item $i \in [r]$. The items $r+1, \ldots, k$ are called *non-dominating items*. The acceptance probabilities of these items are strictly decreasing with increasing rank. However, these probabilities can still be related to corresponding probabilities of dominating items.

Case $k = 2$

We illustrate the general analysis by inspection of the case $k = 2$. Here, there are two possible choices for the parameter $r \in \{1, 2\}$.

If $r = 1$, item 1 is dominating and item 2 is non-dominating. In Sections 3.1 and 3.2 of Paper A, we use a counting argument similar to Section 2.2 to show that

$$p_1 = 2c \cdot \ln(1/c) + c^2 - c - \mathrm{o}(1) \quad \text{and}$$
$$p_2 = 2c \cdot \ln(1/c) + 2c^2 - 2c - \mathrm{o}(1)\,,$$

where the o(1)-terms are asymptotic with respect to n (just as the term $1/n$ in Theorem 2.1). Clearly, these terms vanish in the common asymptotic setting $n \to \infty$. Thus, we ignore them in the competitive ratio from now on.

Table 3.1: Competitive ratio (c.r.) of SINGLE-REF for $k \in [1..10]$ with optimal parameters.

k	1	2	3	4	5	6	7	8	9	10
r	1	1	2	2	2	2	3	3	3	3
c	$1/e$	0.25	0.34	0.29	0.25	0.22	0.28	0.25	0.23	0.21
c.r.	$1/e$	0.41	0.44	0.47	0.49	0.51	0.53	0.54	0.55	0.56

Through Lemma 3.1, the competitive ratio is

$$\frac{p_1 + p_2}{2} = 2c \cdot \ln(1/c) + \frac{3}{2} \cdot (c^2 - c)\,,$$

which attains its maximum of 0.4119 for $c = 0.2545$. Figure 3.1a shows p_1 and p_2 as functions of c as well as the competitive ratio.

If we set $r = 2$, both top-2 items are dominating. From Paper A, Sections 3.1 and 3.2, we obtain

$$p_1 = p_2 = 2c - 3c^2 + c^3 - \mathrm{o}(1)\ .$$

Thus, p_1, p_2, and the competitive ratio coincide. Here, the maximum of 0.3849 is attained for $c = 1 - 1/\sqrt{3} \approx 0.4226$ (see Figure 3.1b).

We conclude that setting $r = 1$ and $c = 0.2545$ maximizes the performance of SINGLE-REF in the case $k = 2$ and yields a 0.4119-competitive algorithm.

3.1.2 Results

The objective in the development of SINGLE-REF was an easy, threshold-based algorithm for the k-secretary problem that beats the barrier of $1/e$ in the competitive ratio already for $k = 2$. A sketch of an analytic proof for $k = 2$ is given below.

In Paper A, we evaluate the performance of SINGLE-REF for larger k in the asymptotic setting $n \to \infty$ and $k = O(1)$. In other words, we assume that the number of items n is large enough that lower order terms vanish, but k is still considered as a constant. The optimal parameters were found by numerical optimization solvers.

Table 3.1 shows the optimal parameters and the resulting competitive ratios of the algorithm for $k \in [10]$. The results also demonstrate that both parameters, the sampling fraction c and the rank r of the reference element, play a crucial role in the fine-tuning of the algorithm for fixed k. For results on a larger scale, see Figure 3.2. This figure depicts the competitive ratios of SINGLE-REF with optimal parameters for $k \in [100]$ and a comparison with the algorithm proposed by Kleinberg [36]. In this interval, SINGLE-REF clearly outperforms the algorithm from [36]. While numerical experiments suggest that this holds true for a much larger interval of k, it is not clear if SINGLE-REF is the superior algorithm for any k.

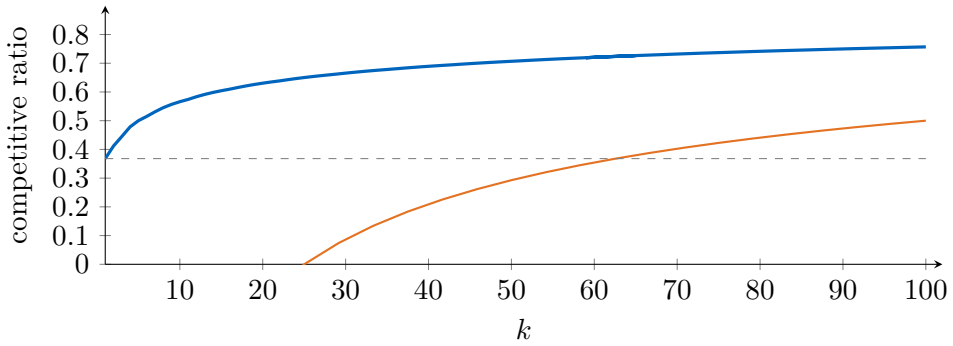


Figure 3.2: Competitive ratio of SINGLE-REF with optimal parameters (blue) in comparison to the algorithm from [36] (orange). The gray line represents the value of $1/e$.

3.2 The Optimistic algorithm

Babaioff et al. [6] proposed two simple algorithms for the k-secretary problem. In Paper A, we revisit the OPTIMISTIC algorithm from [6], which is formally described in Algorithm 3. The algorithm is again based on an initial sampling phase of $t-1$ rounds, with $k+1 \leq t-1 \leq n$, where all items are rejected. However, in contrast to SINGLE-REF, the k best elements from the sampling serve as reference elements. Beginning in round t, the first k items that outperform the current reference element are accepted, where the reference element after ℓ accepted items, for $0 \leq \ell \leq k-1$, is the $(k-\ell)$-th best element from the sampling.

Parameters: $k \in \mathbb{N}$, $t \in [k{+}2..n{+}1]$
In rounds $1, \ldots, t-1$ **do**
  Reject the current item. ▷ `sampling phase`
In rounds $t, \ldots, n$ **do**
  Let s_j (for $1 \leq j \leq k$) be the j-th best profit from the sampling phase.
  Let ℓ be the number of items accepted so far.
  if $\ell < k$ **then**
    Accept the current item if its profit exceeds $s_{k-\ell}$.

Algorithm 3: The OPTIMISTIC algorithm from [6]

Regarding the performance guarantee of OPTIMISTIC, Babaioff et al. [6] stated the following theorem.

Theorem 3.1 ([6]). *For $n \to \infty$ and all $k \geq 1$,* OPTIMISTIC *is $(1/e)$-competitive with $t - 1 = \lfloor n/e \rfloor$.*

However, the authors conjecture that $1/e$ is in fact only a lower bound which is not tight for $k \geq 2$. Moreover, neither the conference paper [6] nor the full version [8] provide a rigorous proof of Theorem 3.1. In Paper A, we present a tight analysis for the case $k = 2$, demonstrating that the competitive ratio exceeds $1/e$ significantly.

Theorem 3.2. *For $n \to \infty$ and $k = 2$,* OPTIMISTIC *is 0.4168-competitive with and $t - 1 = 0.3521n$.*

According to Lemma 3.1, Theorem 3.2 follows immediately from analyzing the probabilities p_1 and p_2. Here, a key lemma is the following.

Lemma 3.2. *The probability that Algorithm 3 for $k = 2$ accepts the second best item equals the probability that Algorithm 1 accepts the best item, assuming that both algorithms sample the same number of rounds.*

In Paper A, we prove this interesting connection between both algorithms by a sophistically tailored bijection between two sets of permutations. This way, for $t - 1 = cn$ with $c \in (0, 1)$, we obtain

$$\begin{aligned} p_1 &= c \cdot \ln(1/c) + c^2 \cdot (1/c - \ln(1/c) - 1) - \mathrm{o}(1)\,, \\ p_2 &= c \cdot \ln(1/c) - \mathrm{o}(1)\,, \end{aligned}$$

yielding a competitive ratio for $n \to \infty$ of

$$\frac{p_1 + p_2}{2} = c \cdot \ln(1/c) + \frac{d^2}{2} \cdot (1/c - \ln(1/c) - 1)\,.$$

By setting $c = 0.3521$, the competitive ratio is maximized and attains a value around 0.4168. We observe that OPTIMISTIC slightly outperforms SINGLE-REF in the case $k = 2$.

Chapter 4

The Knapsack Problem

This chapter deals with the online knapsack problem in the random-order model, formally defined as follows.

> **Problem 4.1 (Online Knapsack under Random Arrival Order)**
> Let $W > 0$ be the capacity of a given resource. For each item $i \in [n]$, let v_i be its profit and s_i its size. The items are presented to the algorithm according to Definition 1.1, whereby n is known to the algorithm. The goal is to find a subset $S \subseteq [n]$, called *packing*, such that $\sum_{i \in S} s_i \leq W$ and $\sum_{i \in S} v_i$ is maximal.

One challenge in the design of online knapsack algorithms is the structural heterogeneity of possible optimal solutions: An optimal packing can be a single item, a set of numerous small items of approximately equal size, or any further collection of items with total size at most W. Therefore, a common algorithmic approach in earlier works [6, 35] is to "guess" an optimal packing as follows: Based on their sizes, items are grouped into two or more classes (e.g., *large* and *small*). The overall algorithm decides for one item class via initial random choice and then performs an algorithm tailored to this class.

In Paper B, we investigate a novel algorithmic approach: Instead of choosing one item class and corresponding algorithm in advance, our algorithm performs two strategies for two item classes sequentially. This *sequential approach* outperforms the *coin tossing approach* from [35] as shown in Section 4.1. To enhance the overall algorithm further, we study a special case where at most two items can be packed, which is described in Section 4.2.

Notation Let I denote the item list and let $\delta \in (0, 1)$ be a parameter specified later. We say that an item i is *large* if $s_i > \delta W$ and *small* if $s_i \leq \delta W$. Let I_L (I_S) denote the list I where small (large) items are replaced by zero-profit large (small) items. Hence, I_L and I_S are essentially obtained from I by projection to large and small items, respectively; however, both lists still contain n items. Let $\mathrm{OPT_L}$ and $\mathrm{OPT_S}$ denote the profits of optimal offline packings for I_L and I_S, respectively. Note that

$$\mathrm{OPT_L} + \mathrm{OPT_S} \geq \mathrm{OPT}\,. \tag{4.1}$$

For simplicity, we assume in the following that for any constant $c \in (0, 1)$, the product cn is integer. For $n \to \infty$, this assumption is without loss.

4.1 Algorithmic framework: sequential vs. coin tossing approach

In this section, we introduce the sequential approach and show that, even in a simpler form than in Paper B, it outperforms the coin tossing approach from [35]. Both approaches make use of an algorithm developed by Kesselheim et al. [35] for the generalized assignment problem with large capacity ratios. Applied to the knapsack problem, we obtain Algorithm 4. We denote this algorithm by $\mathcal{A}_S$ in the following.

Parameters: $d \in (0, 1]$.
In rounds $1, \ldots, dn$ **do**
  Reject all items.
In round $\ell \in \{dn + 1, \ldots, n\}$ **do**
  Let $I_S(\ell) = \{j \in I_S \mid j \text{ arrived before or in round } \ell\}$.
  Let $y_j \in [0, 1]$ be the fraction of item $j \in I_S(\ell)$ in an optimal solution to the fractional knapsack problem for $I_S(\ell)$.
  Pack the current item i with probability y_i if the remaining capacity is large enough.

Algorithm 4: Algorithm $\mathcal{A}_S$ for small items from [35]

The following lemma about $\mathcal{A}_S$ follows from the proof of Theorem 9 in [35].

Lemma 4.1 ([35]). *If the maximum item size in I_S is $(1/2) \cdot W$, we have $\mathrm{E}[\mathcal{A}_S] \geq (3 \cdot (1 - d) - 2 \cdot \ln(1/d)) \cdot \mathrm{OPT_S}$. The maximal competitive ratio of $1 - \ln(9/4)$ is achieved for $d = 2/3$.*

Coin tossing approach

First, we briefly review the approach from [35] on a high level. Here, large and small items are separated according to $\delta = 1/2$. This way, at most one large item can be part of any feasible packing. Hence, the knapsack problem for I_L becomes the secretary problem, as the goal is essentially reduced to packing the most profitable item. This leads to the following algorithm (see Algorithm 5): First, a biased coin is flipped. Depending on the outcome, either one item from I_L is packed using the optimal algorithm for the secretary problem, or (multiple) small items are packed using $\mathcal{A}_S$.

The performance of Algorithm 5 can be evaluated by considering both branches; the choice of λ maximizes the performance in the inferior case.

Theorem 4.1. *The competitive ratio of Algorithm 5 is $\frac{3}{3e+16} > \frac{1}{8.06}$, setting $\lambda = \frac{\beta}{1/e+\beta}$, with $\beta = 1 - \ln(9/4)$, and assuming $n \to \infty$.*

Parameters: $\lambda \in [0, 1]$.
With probability λ **do**
  Apply Algorithm 1 with $t = \lceil n/e \rceil$ on I_L.
With probability $1 - \lambda$ **do**
  Apply $\mathcal{A}_S$ with $d = 2/3$ on I_S.

Algorithm 5: Coin tossing approach from [35]

Proof. Let $\mathrm{E}[\mathcal{A}]$ denote the expected profit of the packing returned by Algorithm 5. By Theorem 2.1 and Lemma 4.1, it holds that

$$\begin{aligned}\mathrm{E}[\mathcal{A}] &\geq \lambda \cdot \frac{1}{e} \cdot \mathrm{OPT_L} + (1-\lambda) \cdot \left(1 - \ln \frac{9}{4}\right) \cdot \mathrm{OPT_S} \\ &\geq \min\left\{\lambda \cdot \frac{1}{e},\ (1-\lambda) \cdot \left(1 - \ln \frac{9}{4}\right)\right\} \cdot (\mathrm{OPT_L} + \mathrm{OPT_S}) \\ &= \frac{\beta}{1 + e\beta} \cdot (\mathrm{OPT_L} + \mathrm{OPT_S})\,.\end{aligned}$$

The claim follows by Equation (4.1) and $\frac{\beta}{1+e\beta} > \frac{1}{8.06}$. □

Sequential approach

One drawback of Algorithm 5 is its disability of adaption after the initial random choice. Suppose that the secretary algorithm on I_L accepts no item during the first $(2/3)n$ rounds. Then, with significant probability, no item will be packed at all. This observation motivates the sequential approach: After a sampling phase, first, an algorithm $\mathcal{A}_L$ runs on I_L, but only within a dedicated time interval. If no item has been packed after this period, the strategy is switched to pack small items from I_S using $\mathcal{A}_S$. A (simplified) description of this algorithmic framework is given in Algorithm 6.

Parameters: $c, d \in (0, 1]$ with $c < d$.
In rounds $1, \ldots, cn$ **do**
  Reject all items.
In rounds $cn + 1, \ldots, dn$ **do**
  Apply $\mathcal{A}_L$ on I_L.
if *the knapsack is empty after round* dn **then**
  In rounds $dn + 1, \ldots, n$ **do**
    Apply $\mathcal{A}_S$ on I_S.

Algorithm 6: Sequential approach

For this subsection, we assume that the algorithm $\mathcal{A}_L$ for large items in Algorithm 6 is Algorithm 1 with $t - 1 = cn$. For the analysis, we first

consider the expected profit of $\mathcal{A}_L$ in the corresponding interval.

Lemma 4.2. *The expected profit returned by $\mathcal{A}_L$ in rounds $cn+1,\dots,dn$ is $\mathrm{E}[\mathcal{A}_L] \geq c \cdot \ln(d/c) \cdot \mathrm{OPT_L}$ assuming $n \to \infty$.*

Proof. The claim can be proven analogous to Theorem 2.1, summing up the probabilities of the events $E_{i,a}$ only for i with $cn \leq i \leq dn$. □

Note that for $d = 1$, Lemma 4.2 implies $\mathrm{E}[\mathcal{A}_L] \geq c \cdot \ln(1/c) \cdot \mathrm{OPT_L}$, which is the bound already shown in Section 2.2. In the framework of Algorithm 6, d will be set to a value smaller than 1 in order to benefit from the subsequent algorithm $\mathcal{A}_S$. This in turn requires that the remaining capacity is large enough. We have the following lemma.

Lemma 4.3. *Let ξ be the event that no item is packed in Algorithm 6 before round $dn+1$. It holds that $\Pr[\xi] = c/d$.*

Proof. The algorithm $\mathcal{A}_L$ packs no item if and only if the maximum profit item of the first dn rounds arrives within rounds $1,\dots,cn$. In a uniformly random permutation, this happens with probability $(cn)/(dn) = c/d$. □

Using the previous two lemmas, we are able to analyze Algorithm 6.

Theorem 4.2. *Algorithm 6 is $(1/6.99)$-competitive for $n \to \infty$ setting $c = 0.52$ and $d = 0.69$.*

Proof. The expected profit of $\mathcal{A}_L$ in rounds $cn+1,\dots,dn$ is $\mathrm{E}[\mathcal{A}_L] \geq c \cdot \ln(d/c) \cdot \mathrm{OPT_L}$ by Lemma 4.2. Let ξ be the event from Lemma 4.3. Conditioned on ξ, all items picked by $\mathcal{A}_S$ during rounds $dn+1,\dots,n$ can be added to the packing, giving expected profit $\mathrm{E}[\mathcal{A}_S \mid \xi] \geq (3 \cdot (1-d) - 2 \cdot \ln(1/d)) \cdot \mathrm{OPT_S}$ by Lemma 4.1. Now, let $\mathrm{E}[\mathcal{A}]$ be the expected profit of the packing returned by Algorithm 6. Using Lemma 4.3 and the previously mentioned bounds, we obtain

$$\begin{aligned}
\mathrm{E}[\mathcal{A}] &= \mathrm{E}[\mathcal{A}_L] + \Pr[\xi] \cdot \mathrm{E}[\mathcal{A}_S \mid \xi] \\
&\geq c \cdot \ln\frac{d}{c} \cdot \mathrm{OPT_L} + \frac{c}{d} \cdot \left(3 \cdot (1-d) - 2 \cdot \ln\frac{1}{d}\right) \cdot \mathrm{OPT_S} \\
&\geq \min\left\{c \cdot \ln\frac{d}{c},\ \frac{c}{d} \cdot \left(3 \cdot (1-d) - 2 \cdot \ln\frac{1}{d}\right)\right\} \cdot (\mathrm{OPT_L} + \mathrm{OPT_S})\,.
\end{aligned}$$

The term $\min\{c \cdot \ln(d/c),\ (c/d) \cdot (3 \cdot (1-d) - 2 \cdot \ln(1/d))\}$ is maximized for $c = 0.52$ and $d = 0.69$ and is larger than $1/6.99$ for these parameters. Applying Equation (4.1) concludes the proof. □

Comparing the results of Theorems 4.1 and 4.2, we observe that the sequential approach outperforms the coin tossing approach, though both algorithms are based on the same algorithmic building blocks.

4.2 The 2-Knapsack problem

A further improvement upon Theorem 4.2 can be obtained by adjusting the definition of large items. By choosing $\delta = 1/3$, the knapsack problem on I_L now consists in finding a maximum-profit packing of cardinality 1 or 2. We call this problem 2-Knapsack.

> **Problem 4.2 (Online 2-Knapsack under Random Arrival Order)**
> The problem is identical to Problem 4.1, but for each item i, it holds that $s_i > (1/3) \cdot W$.

In Paper B, we develop and analyze a simple algorithm for online 2-Knapsack in the random-order model: Using the SINGLE-REF algorithm from Section 3.1 with $k = 2$ and $r = 1$, up to two items of high total profit are identified. While the first item is always packed, the second item can only be packed if the remaining capacity is large enough. This approach establishes the following proposition.

Proposition 4.1. *The 2-Knapsack problem admits an online algorithm of competitive ratio* $1/3.08$ *assuming* $n \to \infty$ *in the random-order model.*

Note that the problem contains the secretary problem as a special case. Thus, no algorithm can have a competitive ratio larger than $1/e < 1/2.71$. We leave the existence of a $(1/e)$-competitive algorithm or a corresponding impossibility result as an open problem.

4.3 Results

The main result of Paper B is a randomized algorithm for Problem 4.1 based on the sequential approach.

Theorem 4.3. *There exists a* $(1/6.65)$*-competitive randomized algorithm for the online knapsack problem in the random-order model assuming* $n \to \infty$*.*

Note that this improves upon the competitive ratio 1/6.99 from the preliminary result of Theorem 4.2. The final algorithm uses the algorithm for the 2-Knapsack problem desribed in Section 4.2 as $\mathcal{A}_L$ instead of Algorithm 1. This way, the input for $\mathcal{A}_S$ is a list of items of maximum size $(1/3) \cdot W$, leading to a stronger performance bound of $\mathcal{A}_S$.

Moreover, we discuss the sequential approach for the generalized assignment problem (GAP) in Section 6 of Paper B. By this approach, we improve the state-of-the-art algorithm for online GAP in the random-order model.

Theorem 4.4. *There exists a* $(1/6.99)$*-competitive randomized algorithm for the online generalized assignment problem in the random-order model assuming* $n \to \infty$*.*

Chapter 5

Bin Packing

Bin packing is a fundamental and widely studied problem in the field of approximation algorithms. In Paper C, we investigate the following online variant of the problem.

> **Problem 5.1 (Online Bin Packing under Random Arrival Order)**
> A list I of n items is presented to the algorithm according to Definition 1.1, where item $i \in [n]$ has size $x_i \in (0, 1]$. The goal is to find a packing into bins of capacity 1 such that the number of used bins is minimized. Formally, a *packing* is an assignment from the set of items to the set of bins such that for any bin, the total size of assigned items does not exceed its capacity.

For offline and online bin packing, two performance measures are widely studied: The *absolute approximation ratio* of an algorithm $\mathcal{A}$, denoted by $R_{\mathcal{A}}$, is the worst-case ratio between $\mathcal{A}(I)$ and $\mathrm{OPT}(I)$ over all lists I; it corresponds to a strict competitive ratio in the case of online algorithms. The *asymptotic approximation ratio* $R^{\infty}_{\mathcal{A}}$ evaluates the same ratio in the limit of $\mathrm{OPT}(I) \to \infty$. Formally,

$$R_{\mathcal{A}} = \sup_{I \in \mathcal{I}} \{\mathcal{A}(I)/\mathrm{OPT}(I)\}\,, \tag{5.1}$$

$$R^{\infty}_{\mathcal{A}} = \limsup_{k \to \infty} \sup_{I \in \mathcal{I}} \{\mathcal{A}(I)/\mathrm{OPT}(I) \mid \mathrm{OPT}(I) = k\}\,, \tag{5.2}$$

where $\mathcal{I}$ is the set of all instances. By replacing the term $\mathcal{A}(I)$ by $\mathrm{E}[\mathcal{A}(I^{\sigma})]$ in Equations (5.1) and (5.2), where the expectation is over the random permutation σ of the input, we obtain the random-order equivalents $RR_{\mathcal{A}}$ and $RR^{\infty}_{\mathcal{A}}$, respectively. This way, the *random-order performance* introduced by Kenyon [32] is precisely $RR^{\infty}_{\mathcal{A}}$. In Paper C, we investigate both $RR_{\mathcal{A}}$ and $RR^{\infty}_{\mathcal{A}}$ for the BEST FIT algorithm.

5.1 The Best Fit algorithm

One of the simplest and earliest algorithms for online bin packing is BEST FIT, introduced by Johnson et al. [29] in 1974. The algorithm packs the current item into the fullest bin where it fits, possibly opening a new bin if no such bin exists. A formal description is given in Algorithm 7.

In the adversarial model, the approximation ratio of BEST FIT is completely understood: Johnson et al. [29] showed that $R^{\infty}_{\mathrm{BF}} = 1.7$, a rather

In rounds $1, \ldots, n$ **do**
Let i be the current item.
Let B be the fullest bin that can accommodate i; if no such bin exists, let B be a new bin.
Pack i into B.

Algorithm 7: The BEST FIT algorithm

recent result by Dósa and Sgall [15] shows that even $R_{\mathrm{BF}} = 1.7$. These bounds are clear adversarial bounds in the sense that both item sizes and arrival order are designed in a way that BEST FIT opens many bins compared to an optimal offline algorithm. This type of construction is illustrated in the following example, which shows a slightly weaker lower bound of 5/3.

Example 5.1. Let $\ell \in \mathbb{N}$, $k = 6\ell$ and $\varepsilon \in (0, 1/12)$. Consider the following list of $n = 3k$ items, where for simplicity, we identify an item with its size:

$$I = \Big(\underbrace{\tfrac{1}{6} - 2\varepsilon, \ldots, \tfrac{1}{6} - 2\varepsilon}_{k}, \ \underbrace{\tfrac{1}{3} + \varepsilon, \ldots, \tfrac{1}{3} + \varepsilon}_{k}, \ \underbrace{\tfrac{1}{2} + \varepsilon, \ldots, \tfrac{1}{2} + \varepsilon}_{k} \Big)$$

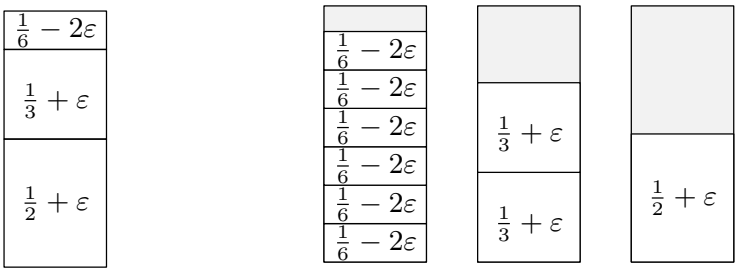
Clearly, an optimal offline algorithm packs one item from each group per bin, using $\mathrm{OPT}(I) = k$ bins in total (see Figure 5.1a). In contrast, it is easily verified that BEST FIT opens $k/6$ bins for the first k items, $k/2$ bins for the next k items, and k bins for the last group (see Figure 5.1b). Thus, $\mathrm{BF}(I) = k/6 + k/2 + k = (5/3) \cdot \mathrm{OPT}(I)$.

The above instance is also a prime example demonstrating that BEST FIT is highly sensitive to the arrival order of items: Assume that I is presented in reversed order to the algorithm, i.e., items arrive in non-increasing order of size. It is easily verified that BEST FIT then produces the optimal packing. In fact, whenever items appear in non-increasing order of size, BEST FIT has an asymptotic approximation ratio of 11/9. This follows directly from the analysis of the offline algorithm BEST FIT DECREASING [29].

5.2 Upper bound for big items under random order

Let us call an item i *large* if $x_i > 1/2$, *medium* if $1/3 < x_i \leq 1/2$, and *small* if $x_i \leq 1/3$.

Even if all items are medium or large, the competitive ratio of BEST FIT under adversarial order is bounded from below by 3/2. This can be seen by removing the small items in Example 5.1. Therefore, large and medium items play a crucial role for the hardness in the general case. On the other hand, we can show that such adversarial inputs are fragile under random order. In Paper C, we therefore study the special case where no small items exist, i.e.



(a) Optimal packing

(b) BEST FIT packing

Figure 5.1: Visualization of Example 5.1

all items are larger than 1/3. Our main contribution is the following upper bound.

Theorem 5.1. *For any list I of items larger than* $1/3$*, we have* $\mathrm{E}[\mathrm{BF}(I^\sigma)] \leq (5/4) \cdot \mathrm{OPT}(I) + 1/2$*, where σ is a permutation drawn uniformly at random.*

The proof of Theorem 5.1 is developed in three main steps which we describe below.

Step 1: Reduction to LM-pairs

We call a bin an *LM-bin* if it contains a large item and a medium item; the corresponding pair of items in an LM-bin of the optimal packing is called *LM-pair*. Without loss of generality, we can assume that the optimal packing has only LM-bins. This way, the problem gets more structured: The approximation guarantee of a packing can be easily measured by the fraction of LM-bins in the packing. In the proof of the above claim, the crucial property is the *monotonicity* of BEST FIT in this case: If all items are larger than 1/3, increasing the size of an item cannot decrease the number of bins used by the algorithm. Interestingly, this property does not hold for BEST FIT in the presence of small items [41].

Step 2: Notion of *good LM-pairs*

Consider the modified version of Example 5.1 where small items are removed: Here, BEST FIT uses $(3/2) \cdot \mathrm{OPT}(I)$ bins if the first k items are medium and the last k items are large. Vice versa, BEST FIT is optimal if all large items arrive before all medium items. In a random permutation, we expect some in-between behavior, which can be quantified by the following definition.

Definition 5.1. Consider a fixed order of items. We say that the LM-pair $\{l_i, m_i\}$ *arrives in good order* if l_i arrives before m_i.

Example 5.2. Let $\{l_i, m_i\}$ for $i \in [4]$ be the i-th LM-bin of an optimal packing. In the item list $I = (m_4, l_2, m_1, l_3, l_4, m_2, l_1, m_3)$, the LM-pairs arriving in good order are $\{l_2, m_2\}$ and $\{l_3, m_3\}$.

The key lemma for the proof of Theorem 5.1 states that the number of LM-bins in the BEST FIT packing is bounded from below by the number of LM-pairs arriving in good order.

Lemma 5.1. *Let I be a list of items larger than $1/3$ and consider a fixed permutation σ'. Let X be the number of LM-pairs arriving in good order in $I^{\sigma'}$. The* BEST FIT *packing for $I^{\sigma'}$ has at least X LM-bins.*

In Paper C, we give an inductive proof of Lemma 5.1 using techniques from bipartite matching.

Step 3: Analyzing a random permutation

In contrast to adversarial order, a positive constant fraction of LM-pairs arrives in good order in a random permutation in expectation.

Lemma 5.2. *Let X denote the number of LM-pairs arriving in good order in a random permutation of k LM-pairs. It holds that* $\mathrm{E}[X] = k/2$.

Proof. The claim follows immediately from the observation that any LM-pair arrives in good order with probability $1/2$ in a random permutation. □

Using the results of Lemmas 5.1 and 5.2, the proof of Theorem 5.1 follows.

Proof of Theorem 5.1. By the reasoning of step 1, we can assume that the optimal packing of I has only LM-bins without loss of generality. Let $k = \mathrm{OPT}(I)$ denote the number of these LM-bins.

Now, we consider the BEST FIT packing. Let Y be a random variable for the number of LM-bins in the packing of BEST FIT for I^σ. Hence, Y large and Y medium items are packed in LM-bins. For the remaining large items, BEST FIT uses $k - Y$ bins. Similarly, at most $\lceil (k-Y)/2 \rceil$ bins are used for medium items (packed in groups of two, except maybe for the last bin). Therefore,

$$\mathrm{BF}(I^\sigma) = Y + (k - Y) + \left\lceil \frac{k-Y}{2} \right\rceil \leq \frac{3k}{2} - \frac{Y}{2} + \frac{1}{2}\,.$$

Now, let X be the number of good order pairs in I^σ. Note that $\mathrm{E}[Y] \geq \mathrm{E}[X] = k/2$ holds through Lemmas 5.1 and 5.2. Using linearity of expectation, we obtain finally

$$\mathrm{E}[\mathrm{BF}(I^\sigma)] \leq \frac{3k}{2} - \frac{\mathrm{E}[Y]}{2} + \frac{1}{2} \leq \frac{3k}{2} - \frac{k/2}{2} + \frac{1}{2} = \frac{5}{4} \cdot \mathrm{OPT}(I) + \frac{1}{2}\,. \qquad \square$$

5.3 Lower bound on asymptotic approximation ratio under random order

As the second main contribution of Paper C, we prove that the asymptotic approximation ratio of BEST FIT in the random-order model is larger than 1.10, improving the previous result $RR_{\mathrm{BF}}^{\infty} \geq 1.08$ from [32].

Theorem 5.2. *It holds that* $RR_{\mathrm{BF}}^{\infty} > 1.10$.

The proof of Theorem 5.2 is based on two pillars. The first pillar is the connection to the *i.i.d.-model*, a related model for stochastic inputs. Here, items are drawn independently and identically distributed (*i.i.d.*) at random. We denote by $I_n(F)$ a list of n items drawn i.i.d. from a discrete distribution F. For a competitive analysis, the online algorithm and an optimal offline algorithm are evaluated in expectation over $I_n(F)$.

The random-order model and the i.i.d.-model are related as follows: As a sequence of i.i.d. items is in uniformly random order by construction, the performance of any algorithm in the random-order model does not decrease for i.i.d. inputs. On the other side, a randomly permuted adversarial input is not necessarily i.i.d. Therefore, hardness results in the i.i.d. model apply to the random-order model as well.

As sampling with replacement is typically easier to analyze than sampling without replacement, several hardness results in the random-order model are based on corresponding hardness results in the i.i.d. model [14, 20, 32]. With respect to our notation and the bin packing problem, we obtain the following lemma.

Lemma 5.3. *Consider any online bin packing algorithm* $\mathcal{A}$. *Let* F *be a discrete distribution and* $I_n(F)$ *a list of* n *i.i.d. samples. For* $n \to \infty$, *there exists a list* I *of* n *items such that*

$$\frac{\mathrm{E}[\mathcal{A}(I^{\sigma})]}{\mathrm{OPT}(I)} \geq \frac{\mathrm{E}[\mathcal{A}(I_n(F))]}{\mathrm{E}[\mathrm{OPT}(I_n(F))]}\,.$$

Moreover, if $X_i \geq c$ *for all* $i \in [n]$ *and a constant* $c > 0$, *it holds that* $\mathrm{OPT}(I) \geq cn$.

By the first part of Lemma 5.3, it is enough to construct a suitable discrete distribution F to obtain a lower bound on RR_{BF}. The second claim implies that for suitable distributions, the term $\mathrm{OPT}(I)$ can be made arbitrarily large. This is necessary to construct a lower bound on $RR_{\mathrm{BF}}^{\infty}$.

Therefore, Theorem 5.2 follows from Lemma 5.3 if a suitable discrete distribution F exists. As stated in the next lemma, a distribution with the desired properties is surprisingly simple.

Lemma 5.4. *Let* F *be the discrete distribution such that an item of size* $1/4$ *has probability* $p = 0.60$ *and an item of size* $1/3$ *has probability* $1 - p$. *For* $n \to \infty$, *it holds that* $\mathrm{E}[\mathrm{BF}(I_n(F))] > (11/10) \cdot \mathrm{E}[\mathrm{OPT}(I_n(F))]$.

To prove Lemma 5.4, the expected numbers of bins used by BEST FIT and an optimal offline algorithm needs to be analyzed. An upper bound on $\mathrm{E}[\mathrm{OPT}(I_n(F)]$ can be obtained easily, as an optimal packing groups items according to their sizes (possibly except for at most two bins). Hence,

$$\mathrm{E}[\mathrm{OPT}(I_n(F))] \leq \frac{np}{4} + \frac{n \cdot (1-p)}{3} + 2\,.$$

Analyzing the behavior of BEST FIT is more elaborate. The key observation is that at any time at most two bins in the packing are open, i.e., can accommodate items in the future. Also, there is a limited number of configurations for open bins. Both properties are due to the fact that the number of item types in F is constant. This way, the behavior of BEST FIT over time can be modeled as an ergodic Markov chain with nine states. The expected total number of bins $\mathrm{E}[\mathrm{BF}(I_n(F))]$ is related to the stationary distribution of the Markov chain.

Chapter 6

Conclusion and Open Problems

This thesis reports the recent progress of [3–5] in understanding three online packing problems in the random-order model. In this model, even simple algorithmic approaches can lead to demanding analyses. Especially for problems admitting constant competitive ratios, it is not uncommon that existing probabilistic analyses are not tight, e.g., due to tail bounds. Regarding the problems considered in this thesis, we see various directions of future work.

In Paper A, we discussed two algorithms for the k-secretary problem. Our analysis of SINGLE-REF seems tight in the setting of $n \to \infty$, however, it remains to characterize the interplay of the parameters k, r, and t formally. A more fundamental question regards the optimal competitive ratio for $k \geq 2$. For ordinal algorithms, the answer can be obtained from the factor-revealing LP approach [12, 13]. In contrast and maybe surprisingly, for general algorithms, this is still an open problem.

Paper B deals with the online knapsack problem under random arrival order. Again, the optimal competitive ratio for this problem is not known and the only impossibility result is inherited from the secretary problem. Thus, it is still open if the problem admits a $(1/e)$-competitive algorithm. Note that the result from [34] on edge-weighted bipartite matching demonstrates that generalizations of the secretary problem may admit $(1/e)$-competitive algorithms by all means. We feel that the consideration of special cases as the 2-Knapsack problem introduced in Paper B may lead to new insights.

In Paper C, we revisited the BEST FIT algorithm for online bin packing under random arrival order. In the general case, the asymptotic approximation ratio of this algorithm is still only bounded by $1.1 < RR^{\infty}_{\mathrm{BF}} \leq 1.5$ and an obvious direction of future work is to close this gap. Moreover, it would be interesting to identify other simple online algorithms for bin packing with a strong performance in the random-order model. For example, is it possible to slightly modify the BEST FIT algorithm such that it significantly beats 1.5 with respect to competitive ratio under random order? Such algorithms seem interesting, both from a theoretical and practical point of view.

Bibliography

[1] G. Aggarwal, R. Motwani, D. Shah, and A. Zhu. Switch scheduling via randomized edge coloring. In *Proc. 44th Symposium on Foundations of Computer Science (FOCS)*, pages 502–512, 2003.

[2] S. Albers and M. Janke. Scheduling in the random-order model. In *Proc. 47th International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 68:1–68:18, 2020.

[3] S. Albers, A. Khan, and L. Ladewig. Improved online algorithms for knapsack and GAP in the random order model. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM)*, volume 145 of *LIPIcs*, pages 22:1–22:23. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

[4] S. Albers, A. Khan, and L. Ladewig. Best fit bin packing with random order revisited. In *Proc. 45th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 170 of *LIPIcs*, pages 7:1–7:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[5] S. Albers and L. Ladewig. New results for the k-secretary problem. In *Proc. 30th International Symposium on Algorithms and Computation (ISAAC)*, volume 149 of *LIPIcs*, pages 18:1–18:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

[6] M. Babaioff, N. Immorlica, D. Kempe, and R. Kleinberg. A knapsack secretary problem with applications. In *Proc. 10th International Workshop on Approximation, Randomization, and Combinatorial Optimization and 11th International Workshop on Randomization and Computation (APPROX/RANDOM)*, pages 16–28, 2007.

[7] M. Babaioff, N. Immorlica, D. Kempe, and R. Kleinberg. Online auctions and generalized secretary problems. *SIGecom Exch.*, 7(2), 2008.

[8] M. Babaioff, N. Immorlica, D. Kempe, and R. Kleinberg. Matroid secretary problems. *Journal of the ACM*, 65(6):35:1–35:26, 2018.

[9] M. Babaioff, N. Immorlica, and R. Kleinberg. Matroids, secretary problems, and online mechanisms. In *Proc. 18th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 434–443, 2007.

[10] B. Bahmani, A. Mehta, and R. Motwani. Online graph edge-coloring in the random-order arrival model. *Theory Comput.*, 8(1):567–595, 2012.

[11] S. Bhattacharya, F. Grandoni, and D. Wajc. Online algorithms for edge coloring via the nibble method. In *Proc. 32nd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2021 (to appear).

[12] N. Buchbinder, K. Jain, and M. Singh. Secretary problems via linear programming. *Math. Oper. Res.*, 39(1):190–206, 2014.

[13] T.-H. H. Chan, F. Chen, and S. H.-C. Jiang. Revealing optimal thresholds for generalized secretary problem via continuous LP: impacts on online K-item auction and bipartite K-matching with random arrival order. In *Proc. 26th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1169–1188, 2015.

[14] M. G. Christ, L. M. Favrholdt, and K. S. Larsen. Online bin covering: Expectations vs. guarantees. *Theor. Comput. Sci.*, 556:71–84, 2014.

[15] G. Dósa and J. Sgall. Optimal analysis of best fit bin packing. In *Proc. 41st International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 429–441, 2014.

[16] E. B. Dynkin. The optimum choice of the instant for stopping a markov process. *Soviet Mathematics*, 4:627–629, 1963.

[17] M. Feldman, O. Svensson, and R. Zenklusen. A simple $O(\log \log(\text{rank}))$-competitive algorithm for the matroid secretary problem. *Math. Oper. Res.*, 43(2):638–650, 2018.

[18] T. S. Ferguson. Who solved the secretary problem? *Statistical science*, 4(3):282–289, 1989.

[19] C. Fischer. *New Results on the Probabilistic Analysis of Online Bin Packing and its Variants*. PhD thesis, Rheinische Friedrich-Wilhelms-Universität Bonn, 2019.

[20] C. Fischer and H. Röglin. Probabilistic analysis of the dual next-fit algorithm for bin covering. In *Proc. 12th Latin American Symposium (LATIN)*, pages 469–482, 2016.

[21] C. Fischer and H. Röglin. Probabilistic analysis of online (class-constrained) bin packing and bin covering. In *Proc. 13th Latin American Symposium (LATIN)*, volume 10807 of *Lecture Notes in Computer Science*, pages 461–474. Springer, 2018.

[22] P.R. Freeman. The secretary problem and its extensions: A review. *International Statistical Review/Revue Internationale de Statistique*, pages 189–206, 1983.

[23] J. P. Gilbert and F. Mosteller. Recognizing the maximum of a sequence. *Journal of the American Statistical Association*, 61(313):35–73, 1966.

[24] O. Göbel, T. Kesselheim, and A. Tönnis. Online appointment scheduling in the random order model. In *Proc. 23rd Annual European Symposium on Algorithms (ESA)*, pages 680–692, 2015.

[25] R. L. Graham. Bounds on multiprocessing timing anomalies. *SIAM Journal of Applied Mathematics*, 17(2):416–429, 1969.

[26] R. L. Graham, D. E. Knuth, and O. Patashnik. *Concrete Mathematics: A Foundation for Computer Science, 2nd Ed.* Addison-Wesley, 1994.

[27] A. Gupta and S. Singla. Random-order models. In T. Roughgarden, editor, *Beyond the Worst-Case Analysis of Algorithms*, chapter 11. Cambridge University Press, 2020 (to appear).

[28] D. S. Johnson. Fast algorithms for bin packing. *J. Comput. Syst. Sci.*, 8(3):272–314, 1974.

[29] D. S. Johnson, A. J. Demers, J. D. Ullman, M. R. Garey, and R. L. Graham. Worst-case performance bounds for simple one-dimensional packing algorithms. *SIAM J. Comput.*, 3(4):299–325, 1974.

[30] E. G. Coffman Jr., J. Csirik, L. Rónyai, and A. Zsbán. Random-order bin packing. *Discret. Appl. Math.*, 156(14):2810–2816, 2008.

[31] R. M. Karp, U. V. Vazirani, and V. V. Vazirani. An optimal algorithm for on-line bipartite matching. In *Proc. 22nd Annual ACM Symposium on Theory of Computing (STOC)*, pages 352–358, 1990.

[32] C. Kenyon. Best-fit bin-packing with random order. In *Proc. 7th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 359–364, 1996.

[33] T. Kesselheim, R. D. Kleinberg, and R. Niazadeh. Secretary problems with non-uniform arrival order. In *Proc. 47th Annual ACM Symposium on Theory of Computing (STOC)*, pages 879–888, 2015.

[34] T. Kesselheim, K. Radke, A. Tönnis, and B. Vöcking. An optimal online algorithm for weighted bipartite matching and extensions to combinatorial auctions. In *Proc. 21st Annual European Symposium on Algorithms (ESA)*, pages 589–600, 2013.

[35] T. Kesselheim, K. Radke, A. Tönnis, and B. Vöcking. Primal beats dual on online packing LPs in the random-order model. *SIAM J. Comput.*, 47(5):1939–1964, 2018.

[36] R. D. Kleinberg. A multiple-choice secretary algorithm with applications to online auctions. In *Proc. 16th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 630–631, 2005.

[37] D. V. Lindley. Dynamic programming and decision theory. *Applied Statistics*, pages 39–51, 1961.

[38] M. Mahdian and Q. Yan. Online bipartite matching with random arrivals: an approach based on strongly factor-revealing LPs. In *Proc. 43rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 597–606, 2011.

[39] A. Meyerson. Online facility location. In *Proc. 42nd IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, pages 426–431, 2001.

[40] M. Molinaro. Online and random-order load balancing simultaneously. In *Proc. 28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1638–1650, 2017.

[41] F. D. Murgolo. Anomalous behavior in bin packing algorithms. *Discret. Appl. Math.*, 21(3):229–243, 1988.

[42] D. Naori and D. Raz. Online multidimensional packing problems in the random-order model. In *Proc. 30th International Symposium on Algorithms and Computation (ISAAC)*, pages 10:1–10:15, 2019.

[43] C. J. Osborn and E. Torng. List's worst-average-case or WAC ratio. *J. Sched.*, 11(3):213–215, 2008.

[44] D. D. Sleator and R. E. Tarjan. Amortized efficiency of list update and paging rules. *Commun. ACM*, 28(2):202–208, 1985.

[45] Y. Zhou, D. Chakrabarty, and R. M. Lukose. Budget constrained bidding in keyword auctions and online knapsack problems. In *Proc. 4th International Workshop Internet and Network Economics (WINE)*, pages 566–576, 2008.

Appendix A

New Results for the k-Secretary Problem

Bibliographic information S. Albers and L. Ladewig. New results for the k-secretary problem. In *Proc. 30th International Symposium on Algorithms and Computation (ISAAC)*, volume 149 of *LIPIcs*, pages 18:1–18:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

Summary In this paper, we investigate the k-secretary problem, a natural generalization of the classic secretary problem with cardinality constraint $k \in \mathbb{N}$ instead of one. Our focus lies on the relevant case where k is small. We study simple threshold-based algorithms in the style of the optimal $(1/e)$-competitive algorithm for the secretary problem. The main contribution is a new algorithm SINGLE-REF, which is based on the natural approach of accepting the first k items that beat a reference element after an initial sampling phase. We explicitly compute its competitive ratios for $1 \leq k \leq 100$, showing that the algorithm is 0.4119-competitive already at $k = 2$, which significantly beats the barrier of $1/e$. Moreover, we revisit the OPTIMISTIC algorithm proposed by Babaioff et al. [APPROX'07] for which no rigorous analysis is known. For the case $k = 2$, we present a thorough analysis based on a non-trivial combinatorial property of this algorithm. It turns out that OPTIMISTIC is 0.4168-competitive.

Individual contributions

- Initial proposal of the SINGLE-REF algorithm, development of the analyses and numerical optimization of parameters in Section 3
- Observation of Lemma 4.1 and development of the proofs in Section 4
- Composition of the manuscript including all technical and non-technical parts (refined based on discussions with co-author)

New Results for the k-Secretary Problem

Susanne Albers
Department of Informatics, Technical University of Munich, Germany
albers@in.tum.de

Leon Ladewig
Department of Informatics, Technical University of Munich, Germany
ladewig@in.tum.de

Abstract

Suppose that n numbers arrive online in random order and the goal is to select k of them such that the expected sum of the selected items is maximized. The decision for any item is irrevocable and must be made on arrival without knowing future items. This problem is known as the *k-secretary problem*, which includes the classical secretary problem with the special case $k = 1$. It is well-known that the latter problem can be solved by a simple algorithm of competitive ratio $1/e$ which is asymptotically optimal. When k is small, only for $k = 2$ does there exist an algorithm beating the threshold of $1/e$ [Chan et al. SODA 2015]. The algorithm relies on an involved selection policy. Moreover, there exist results when k is large [Kleinberg SODA 2005].

In this paper we present results for the k-secretary problem, considering the interesting and relevant case that k is small. We focus on simple selection algorithms, accompanied by combinatorial analyses. As a main contribution we propose a natural deterministic algorithm designed to have competitive ratios strictly greater than $1/e$ for small $k \geq 2$. This algorithm is hardly more complex than the elegant strategy for the classical secretary problem, optimal for $k = 1$, and works for all $k \geq 1$. We explicitly compute its competitive ratios for $2 \leq k \leq 100$, ranging from 0.41 for $k = 2$ to 0.75 for $k = 100$. Moreover, we show that an algorithm proposed by Babaioff et al. [APPROX 2007] has a competitive ratio of 0.4168 for $k = 2$, implying that the previous analysis was not tight. Our analysis reveals a surprising combinatorial property of this algorithm, which might be helpful for a tight analysis of this algorithm for general k.

2012 ACM Subject Classification Theory of computation → Online algorithms

Keywords and phrases Online algorithms, secretary problem, random order model

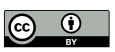
Digital Object Identifier 10.4230/LIPIcs.ISAAC.2019.18

Funding Work supported by the European Research Council, Grant Agreement No. 691672.

1 Introduction

The *secretary problem* is a well-known problem in the field of optimal stopping theory and is defined as follows: Given a sequence of n numbers which arrive online and in random order, select the maximum number. Thereby, upon arrival of an item, the decision to accept or reject it must be made immediately and irrevocably, especially without knowing future items. The statement of the problem dates back to the 1960s and its solution is due to Lindley [23] and Dynkin [10]. For discussions on the origin of the problem, we refer to the survey [13].

In the past years, generalizations of the secretary problem involving selection of multiple items have become very popular. We consider one of the most canonical generalizations known as the *k-secretary problem*: The algorithm is allowed to choose k elements and the goal is to maximize the expected sum of accepted elements. Other objective functions, such as maximizing the probability of accepting the k best [2, 14] or general submodular functions [20], have been studied as well. Maximizing the sum of accepted items is closely related to the *knapsack secretary problem* [3, 19]. If all items have unit weight and thus the knapsack capacity is a cardinality bound, the k-secretary problem arises. The *matroid*



secretary problem, introduced by Babaioff et al. [6], is a generalization where an algorithm must maintain a set of accepted items that form an independent set of a given matroid. We refer the reader to [11, 12, 22] for recent work. If the matroid is k-uniform, again, the k-secretary problem occurs. Another closely related problem was introduced by Buchbinder, Jain, and Singh [8]. In the *(J, K)-secretary problem*, an algorithm has J choices and the objective is to maximize the number of selected items among the K best. Assuming the ordinal model [17] and a monotonicity property of the algorithm, any c-competitive algorithm for the (k, k)-secretary problem is c-competitive for the k-secretary problem, and vice versa [8]. In the ordinal model [17], an algorithm decides based on the total order of items only, rather than on their numeric values. In fact, most known and elegant algorithms for the k-secretary problem assume the ordinal model [3, 10, 21, 23].

The large interest in generalizations of the classical secretary problem is motivated mainly by numerous applications in online market design [4, 6, 21]. Apart from these applications, the secretary problem is the prototype of an online problem analyzed in the random order model: An adversarial input order often rules out (good) competitive ratios when considering online optimization problems without further constraints. By contrast, the assumption that the input is ordered randomly improves the competitive ratios in many optimization problems. This includes packing problems [18, 19], scheduling problems [15], and graph problems [7, 24]. Therefore, developing new techniques for secretary problems may, more generally, yield relevant insights for the analysis of online problems in randomized input models as well.

1.1 Previous Work

The k-secretary problem was introduced by Kleinberg [21] in 2005. He presents a randomized algorithm attaining a competitive ratio of $1-5/\sqrt{k}$, which approaches 1 for $k \to \infty$. Moreover, Kleinberg gives in [21] a hardness result stating that any algorithm has a competitive ratio of $1 - \Omega(\sqrt{1/k})$. Therefore, from an asymptotic point of view, the k-secretary problem is solved by Kleinberg's result. However, the main drawback can be seen in the fact that the competitive ratio is not defined if $k \leq 24$ and breaks the barrier of $1/e$ only if $k \geq 63$ (see Figure 2, p. 11).

In 2007 the problem was revisited by Babaioff et al. [3]. The authors propose two algorithms called VIRTUAL and OPTIMISTIC and prove that both algorithms have a competitive ratio of at least $1/e$ for any k. While the analysis of VIRTUAL is simple and tight, it takes much more effort to analyze OPTIMISTIC [3, 4]. The authors believe that their analysis for OPTIMISTIC is not tight for $k \geq 2$.

Buchbinder, Jain, and Singh [8] developed a framework to analyze secretary problems and their optimal algorithms using linear programming techniques. By numerical simulations for the (k, k)-secretary problem with $n = 100$, Buchbinder et al. obtained competitive ratios of 0.474, 0.565, and 0.612, for $k = 2, 3$, and 4, respectively. However, obtaining an algorithm from their framework requires a formal analysis of the corresponding LP in the limit of $n \to \infty$, which is not provided in the article [8, p. 192].

Chan, Chen, and Jiang [9] revisited the (J, K)-secretary problem and obtained several fundamental results. Notably, they showed that optimal algorithms for the k-secretary problem require access to the numeric values of the items, which complements the previous line of research in the ordinal model. Chan et al. demonstrate this by providing a 0.4920-competitive algorithm for the 2-secretary problem which is based on a 0.4886-competitive algorithm for the $(2, 2)$-secretary problem. Still, an analysis for the general (J, K)-case is not known, even for $J = K$. Moreover, the resulting algorithms seem overly involved. This dims the prospect of elegant k-secretary algorithms for $k \geq 3$ obtained from this approach.

Table 1 Competitive ratios α of SINGLE-REF for $k \in [1..20]$.

k	1	2	3	4	5	6	7	8	9	10
α	$1/e$	0.4119	0.4449	0.4785	0.4999	0.5148	0.5308	0.5453	0.5567	0.5660
k	11	12	13	14	15	16	17	18	19	20
α	0.5740	0.5834	0.5914	0.5983	0.6043	0.6096	0.6155	0.6211	0.6261	0.6306

1.2 Our Contribution

We study the k-secretary problem, the most natural and immediate generalization of the classical secretary problem. While the extreme cases $k = 1$ and $k \to \infty$ are well studied, hardly any results for small values of $k \geq 2$ exist. We believe that simple selection algorithms, performing well for small k, are interesting both from a theoretical point of view and for practical settings. Moreover, the hope is that existing algorithms for related problems based on k-secretary algorithms can be improved this way [8, p. 191]. We study algorithms designed for the ordinal model, which guarantees robustness and plainer decision rules.

For this purpose, we propose a simple deterministic algorithm SINGLE-REF. This algorithm uses a single value as threshold for accepting items. Although similar approaches based on this natural idea have been used to solve related problems [1], to the best of our knowledge, this algorithm has not been explored for the k-secretary problem so far. As a strength of our algorithm we see its simplicity: It is of plain combinatorial nature and can be fine-tuned using only two parameters. In contrast, the optimal algorithms which follow theoretically from the (J, K)-secretary approach [9] would involve k^2 parameters and the same number of different decision rules.

An important insight for the analysis of SINGLE-REF is that items can be partitioned into two classes, which we will call *dominating* and *non-dominating*. Both have certain properties on which we base our fully parameterized analysis. In Table 1, we list the competitive ratios of SINGLE-REF for $k \leq 20$. While the competitive ratio for $k = 1$ is optimal, we obtain a value significantly greater than $1/e$ already for $k = 2$. Furthermore, the competitive ratios are monotonically increasing in the interval $k \in [1..20]$, already breaking the threshold of 0.5 at $k = 6$. Numerical computations suggest that this monotonicity holds for general k. See Figure 2 (p. 11) for the competitive ratios up to $k = 100$ and a comparison with Kleinberg's algorithm [21]. Providing a closed formula for the competitive ratio for any value of k is one direction of future work (see Section 5).

Moreover, we investigate the OPTIMISTIC algorithm by Babaioff et al. [3] for the case $k = 2$. Although Chan et al. [9] provide the optimal algorithm for $k = 2$, we think studying this elegant algorithm is interesting for two reasons: First, a tight analysis of OPTIMISTIC is stated as open problem in [3]. Article [3] does not provide the proof of the $(1/e)$-bound and a recent journal publication [5] (evolved from [3] and [6]) does not cover the OPTIMISTIC algorithm at all. We make progress in this problem by proving that for $k = 2$ its competitive ratio is exactly 0.4168 which significantly breaks the $(1/e)$-barrier. Second, our proof reveals an interesting property of this algorithm, which we show in Lemma 4.1: The probability that OPTIMISTIC accepts the second best item is exactly the probability that the optimal algorithm for $k = 1$ from [10, 23] accepts the best item. A similar property might hold for $k \geq 3$, which could be a key insight into the general case.

From a technical point of view, we derive the exact probabilities using basic combinatorial constructs exclusively. This is in contrast to previous approaches [8, 9] which can only be analyzed using heavyweight linear programming techniques. In addition, we always

consider the asymptotic setting of $n \to \infty$ items, which gives more meaningful bounds on the competitive ratio. Throughout the analyses of both algorithms, we associate probabilities with sets of permutations (see Section 2.2). Hence, probability relations can be shown equivalently by set relations. This is a simple but powerful technique which may be useful in the analysis of other optimization problems with random arrival order as well.

2 Preliminaries

Let $v_1 > v_2 > \ldots > v_n$ be the *elements* (also called *items*) of the input. In the ordinal model, we can assume w.l.o.g. all items to be distinct. Therefore we say that i is the *rank* of element v_i. An *input sequence* is any permutation of the list $v_1, \ldots, v_n$. We denote the position of an element v given a specific input sequence π with $\text{pos}_\pi(v) \in \{1, \ldots, n\}$ and write $\text{pos}(v)$ whenever the input sequence is clear from the context.

Given any input sequence, an algorithm can accept up to k items, where the decision whether to accept or reject an item must be made immediately upon its arrival. Let ALG denote the sum of items accepted by the algorithm. The algorithm is *α-competitive* if $\mathbf{E}[\text{ALG}] \geq \alpha \cdot \text{OPT}$ holds for all item sets. Here the expectation is taken over the uniform distribution of all $n!$ input sequences and $\text{OPT} = \sum_{i=1}^{k} v_i$.

Notation. For $a, b \in \mathbb{N}$ with $a \leq b$, we use the notation $[a..b]$ to denote the set of integers $\{a, a+1, \ldots, b\}$ and write $[a]$ for $[1..a]$. The (half-)open integer intervals $(a..b]$, $[a..b)$, and $(a..b)$ are defined accordingly. Further, we use the notation $n^{\underline{k}}$ for the falling factorial $\frac{n!}{(n-k)!}$.

2.1 Algorithms

In the following, we state the OPTIMISTIC algorithm proposed by Babaioff et al. (Algorithm 1) and our proposed algorithm SINGLE-REF (Algorithm 2) and compare both strategies.

Algorithm 1 OPTIMISTIC [3].

Parameters: $t \in (k..n-k]$ (sampling threshold)
1 **Sampling phase**: Reject the first $t-1$ items.
2 Let $s_1 > \ldots > s_k$ be the k best items from the sampling phase.
3 **Selection phase:** As j-th accepted item, choose the first item better than s_{k-j+1}.

Algorithm 2 SINGLE-REF.

Parameters: $t \in (k..n-k]$ (sampling threshold), $r \in [k]$ (reference rank)
1 **Sampling phase**: Reject the first $t-1$ items.
2 Let s_r be the r-th best item from the sampling phase.
3 **Selection phase:** Choose the first k items better than s_r.

While both algorithms consist of a sampling phase in which the first $t-1$ items are rejected, the main difference is the policy for accepting items: OPTIMISTIC uses the k best items from the sampling as reference elements. Right after the sampling phase, the first item better than s_k (the k-th best from the sampling) will be accepted. The following accepted items are chosen similarly, but with $s_{k-1}, s_{k-2}, \ldots, s_1$ as reference items. Note that this algorithm always sticks to this order of reference points, even if the first item already outperforms s_1. Hence, it is optimistic in the sense that it always expects that high-value items occur in the future.

SINGLE-REF has a simpler structure since it only uses a single item s_r from the sampling as reference point. Here, each item is compared to s_r (the r-th best from sampling), thus the first k elements better than s_r will be selected. Despite its simpler structure, the analysis of SINGLE-REF is involved due to the additional parameter r, as it is not clear how to choose this parameter optimally.

Note that in the case $k = 1$, OPTIMISTIC and SINGLE-REF (when setting $r = 1$) become the strategy known for the classical secretary problem [10, 23]: After rejecting the first $t - 1$ items, choose the first one better than the best from sampling. A simple argument shows that this strategy selects the best item with probability $\frac{t-1}{n} \sum_{i=t}^{n} \frac{1}{i-1}$. If n tends to infinity and $t - 1 \approx n/e$, this term approaches $1/e$ which is optimal.

The following lemma is used to bound the competitive ratios of both algorithms. It heavily relies on the monotonicity property of the algorithms, i.e., for any $v_i > v_j$, both algorithms select v_i with greater or equal probability than v_j.

▶ **Lemma 2.1.** *Let $\mathcal{A}$ be* OPTIMISTIC *or* SINGLE-REF *and for each $i \in [n]$ let p_i be the probability that $\mathcal{A}$ selects item v_i. The competitive ratio of $\mathcal{A}$ is $(1/k) \sum_{i=1}^{k} p_i$.*

Proof. First, we will argue that $p_i \geq p_{i+1}$ for all $i \in [n-1]$, i.e., $\mathcal{A}$ selects items of smaller rank with greater or equal probability. This follows if we can show that the number of permutations where v_{i+1} is accepted is not greater than the respective number of permutations for v_i (this concept is described more detailed in Section 2.2).

Consider any input sequence π in which v_{i+1} is accepted. Let $s_j < v_{i+1}$ be the sampling item to which v_{i+1} is compared (in case of SINGLE-REF we have $j = r$). Since v_{i+1} is accepted, we have $s_j \neq v_i$. By swapping v_i with v_{i+1}, we obtain a new permutation π' with the same reference element s_j. This is obvious if v_i is not in the sampling of π. Otherwise, note that in the ordered sequences of sampling items from π and π', both v_{i+1} and v_i have the same position. This implies that s_j is the j-th best sampling item in π'. Further, item v_i is at the former position of v_{i+1} in π', thus $\mathcal{A}$ accepts v_i at this position since $v_i > v_{i+1} > s_j$.

Thus, both sequences $p_1, \ldots, p_k$ and $v_1, \ldots, v_k$ are sorted decreasingly. Let $\mathrm{OPT}_k = \sum_{i=1}^{k} v_i$ and $\mathbf{E}[\mathcal{A}]$ be the expected sum of the items accepted by $\mathcal{A}$. Chebyshev's sum inequality [16] states that if $a_1 \geq a_2 \geq \ldots \geq a_n$ and $b_1 \geq b_2 \geq \ldots \geq b_n$, then $\sum_{i=1}^{n} a_i b_i \geq (1/n) \left(\sum_{i=1}^{n} a_i\right) \left(\sum_{i=1}^{n} b_i\right)$. Applying this inequality yields

$$\mathbf{E}[\mathcal{A}] = \sum_{i=1}^{n} p_i v_i \geq \sum_{i=1}^{k} p_i v_i \geq \frac{1}{k} \left(\sum_{i=1}^{k} v_i\right) \left(\sum_{i=1}^{k} p_i\right) = \left(\frac{1}{k} \sum_{i=1}^{k} p_i\right) \mathrm{OPT}_k \,.$$

Note that the above inequalities are tight: Assuming that the first k items are almost identical, i.e. $v_i = 1 - i\varepsilon$ for $i \in [1..k]$ and $\varepsilon \to 0$, and $v_i = 0$ for all remaining items of rank $i \in (k..n]$, the competitive ratio is exactly $(1/k) \sum_{i=1}^{k} p_i$. ◀

The same argument is used in [8] to show the equivalence of the k-secretary and the (k, k)-secretary problem for ordinal monotone algorithms.

2.2 Random Order Model

To analyze an algorithm given a random permutation, we often fix an order $u_1, u_2, \ldots, u_n$ of positions. Then, we draw the element for position u_1 uniformly from all n elements, next the element for position u_2 from the remaining $n - 1$ elements, and so on. It is easy to see that by this process we obtain a permutation drawn uniformly at random.

Moreover, the uniform distribution allows us to prove probability relations using functions: Suppose that p_i is the probability that item v_i is accepted in a random permutation, then $p_i = |P_i| / n!$ where P_i is the set of all input sequences where v_i is accepted. Thus, we can

Table 2 Several identities involving binomial coefficients [16].

Rule	Equation	Parameters
(R1) Sum of products	$\sum_{k=0}^{l} \binom{l-k}{m}\binom{q+k}{n} = \binom{l+q+1}{m+n+1}$	$l, m, n, q \in \mathbb{Z}$ with $l, m \geq 0$ and $n \geq q \geq 0$
(R2) Symmetry	$\binom{n}{k} = \binom{n}{n-k}$	$n, k \in \mathbb{Z}$ with $n \geq 0$
(R3) Trinomial revision	$\binom{r}{m}\binom{m}{k} = \binom{r}{k}\binom{r-k}{m-k}$	$m, k \in \mathbb{Z}$ and $r \in \mathbb{R}$

prove $p_i \leq p_j$ by finding an injective function $f\colon P_i \to P_j$ and get $p_i = p_j$ if f is bijective. For example, this technique turns out to be highly useful in the proof of Lemma 4.1, where probabilities of different algorithms are related.

2.3 Combinatorics

We often need to analyze probabilities described by the following random experiment.

▶ **Fact 2.2.** *Suppose there are N balls in an urn from which M are blue and $N - M$ red. The probability of drawing K blue balls without replacement in a sequence of length K is* $h(N, M, K) := \binom{M}{K} / \binom{N}{K}$.

This fact follows from a special case of the hypergeometric distribution.

Furthermore, we make use of several identities involving binomial coefficients throughout the following sections. These equations, denoted by (R1), (R2), and (R3), are listed in Table 2.

3 Analysis of SINGLE-REF

In this section we analyze our proposed algorithm SINGLE-REF, which we denote by $\mathcal{A}$ throughout this section. Recall that this algorithm uses s_r, the r-th best sampling item, as the threshold for accepting items. As implied by the proof of Lemma 2.1, only the k largest items $v_1, \ldots, v_k$ contribute to the objective function. One essential idea of our approach is to separate the set of top-k items into two classes according to the following definition.

▶ **Definition 3.1.** *We say that item v_i is dominating if $i \leq r$, and non-dominating if $r + 1 \leq i \leq k$.*

The crucial property of dominating items becomes clear in the following scenario: Assume that any dominating item v occurs after the sampling phase. Since s_r is the r-th best item from the sampling phase, it follows that $v > s_r$. That is, each dominating item outside the sampling beats the reference item. Therefore there are only two situations when dominating items are rejected: Either they appear before position t, or after k accepted items.

3.1 Acceptance of Dominating Items

First we focus on dominating items. As we will show in Lemma 3.2, the algorithm cannot distinguish between them and thus each dominating item has equal acceptance probability.

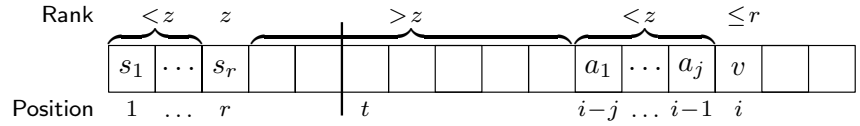


Figure 1 Event $\tilde{E}_j(z,i)$ considered in the proof of Lemma 3.2.

▶ **Lemma 3.2.** *Let v be a dominating item and $j \in [0..k)$. Let E_j be the event that $\mathcal{A}$ selects v as $(j+1)$-th item. It holds that* $\mathbf{Pr}[E_j] = \frac{\kappa\tau}{n} \sum_{i=t+j}^{n} \binom{i-t}{j} \frac{1}{(i-1)^{\underline{r+j}}}$, *where $\tau = (t-1)^{\underline{r}}$ and $\kappa = (r-1+j)^{\underline{j}}$.*

Proof. Let $E_j(z,i)$ be the event that $\mathcal{A}$ accepts v as $(j+1)$-th item at position $i = \mathrm{pos}(v)$ and s_r has rank z (thus $s_r = v_z$). Note that there must be elements $s_1, \ldots, s_{r-1}$ of rank smaller than z in the sampling (such that s_r is in fact the r-th best sampling element). Similarly, there must be j elements $a_1, \ldots, a_j$ after the sampling but before v of rank smaller than z (which are accepted by $\mathcal{A}$).

The proof is in several steps. We first consider a stronger event $\tilde{E}_j(z,i)$. Later, we show how the probability of $E_j(z,i)$ can be obtained from $\tilde{E}_j(z,i)$. In the end, the law of total probability yields $\mathbf{Pr}[E_j]$.

Analysis of $\tilde{E}_j(z,i)$. Event $\tilde{E}_j(z,i)$ is defined as $E_j(z,i)$ with additional position constraints (see Figure 1): Elements $s_1, \ldots, s_r$ are in this order at the first r positions and elements $a_1, \ldots, a_j$ are in this order at the j positions immediately before v. Therefore, $\tilde{E}_j(z,i)$ occurs if and only if the following conditions hold:

- **(i)** $\mathrm{pos}(v) = i$, $\mathrm{pos}(s_\ell) = \ell$ for $\ell \in [r]$, and $\mathrm{pos}(a_m) = i - j + m - 1$ for $m \in [j]$.
- **(ii)** Elements $s_1, \ldots, s_{r-1}$ have rank smaller than z
- **(iii)** Elements $a_1 \ldots, a_j$ have rank smaller than z
- **(iv)** All remaining items at positions $r+1, \ldots, i-j-1$ have rank greater than z.

Using the concept described in Section 2.2, we think of sequentially drawing the elements for the positions $1, \ldots, r, i-j, \ldots, i$ and then $r+1, \ldots, i-j-1$. The probability for (i) is $\prod_{\ell=0}^{j+r} \frac{1}{n-\ell} = 1/n^{\underline{j+r+1}} =: \beta$, since each item has the same probability to occur at each remaining position. In (ii), the $r-1$ elements can be chosen out of $z-2$ remaining items of rank smaller than z (since v is dominating and was already drawn). Therefore we get a factor of $\binom{z-2}{r-1}$. After this step, there remain $z-2-(r-1) = z-r-1$ elements of rank smaller than z, so we get factor $\binom{z-r-1}{j}$ for step (iii).
Finally, the probability of (iv) can be formulated using Fact 2.2. Note that at this point, there remain $n-(1+r+j)$ items and no item of rank greater than z has been drawn so far. In terms of the random experiment described in Fact 2.2, we draw $K = i-j-r-1$ balls (items) from an urn of size $N = n-(1+r+j)$ where $M = n-z$ balls are blue (rank greater than z). Hence, the probability for (iv) is $H := h(n-r-j-1, n-z, i-j-r-1)$. Therefore we obtain

$$\mathbf{Pr}\left[\tilde{E}_j(z,i)\right] = \beta \cdot \binom{z-2}{r-1}\binom{z-r-1}{j} \cdot H\,. \tag{1}$$

This term can be simplified further by applying (R3) and (R2). Let $R = z-2$, $K = r-1$, and $M = j+r-1$. It holds that

$$\binom{z-2}{r-1}\binom{z-r-1}{j} \overset{(R3)}{=} \binom{R}{M}\binom{M}{K} \overset{(R2)}{=} \binom{R}{M}\binom{M}{M-K} = \binom{z-2}{j+r-1}\binom{j+r-1}{j}.$$

Let $\kappa = (j+r-1)^{\underline{j}}$, then $\binom{j+r-1}{j} = \kappa/j!$ and we get $\mathbf{Pr}\left[\tilde{E}_j(z,i)\right] = \frac{\beta\kappa}{j!} \cdot \binom{z-2}{j+r-1} \cdot H\,.$

Relating $\tilde{E}_j(z,i)$ to $E_j(z,i)$. In contrast to $\tilde{E}_j(z,i)$, in the event $E_j(z,i)$, the elements $s_1,\dots,s_r$ can have any positions in $[t-1]$ and $a_1\dots,a_j$ any positions in $[t..i)$. In the random order model, the probability of an event depends linearly on the number of permutations for which the event happens. Hence, we can multiply the probability with corresponding factors $(t-1)^{\underline{r}} =: \tau$ and $(i-t)^{\underline{j}} = \binom{i-t}{j} j!$ and get $\mathbf{Pr}[E_j(z,i)] = \binom{i-t}{j}\tau j! \cdot \mathbf{Pr}\left[\tilde{E}_j(z,i)\right]$.

Relating $E_j(z,i)$ to E_j. As the final step, we sum over all possible values for i and z to obtain $\mathbf{Pr}[E_j]$. The position i of item v ranges between $t+j$ and n, while the reference rank z is between $r+j+1$ (there are $r-1$ sampling elements and $j+1$ accepted elements of rank less than z) and n. Thus we get:

$$\begin{aligned}\mathbf{Pr}[E_j] &= \sum_{i=t+j}^{n}\sum_{z=r+j+1}^{n}\mathbf{Pr}[E_j(z,i)] = \tau j!\sum_{i=t+j}^{n}\binom{i-t}{j}\sum_{z=r+j+1}^{n}\mathbf{Pr}\left[\tilde{E}_j(z,i)\right]\\ &= \beta\kappa\tau\sum_{i=t+j}^{n}\binom{i-t}{j}\sum_{z=r+j+1}^{n}\binom{z-2}{j+r-1}\cdot H\\ &= \beta\kappa\tau\sum_{i=t+j}^{n}\binom{i-t}{j}\frac{1}{\binom{n-r-j-1}{i-j-r-1}}\sum_{z=r+j+1}^{n}\binom{z-2}{j+r-1}\binom{n-z}{i-j-r-1}, \qquad (2)\end{aligned}$$

where the last step follows from Fact 2.2. The sum over z in Equation (2) can be resolved using (R1). Let $L = n-r-j-1$, $N = Q = r+j-1$, and $M = i-j-r-1$. Then we have

$$\begin{aligned}&\sum_{z=r+j+1}^{n}\binom{z-2}{j+r-1}\binom{n-z}{i-j-r-1} = \sum_{z=0}^{n-r-j-1}\binom{r+j-1+z}{j+r-1}\binom{n-r-j-1-z}{i-j-r-1}\\ &= \sum_{z=0}^{L}\binom{Q+z}{N}\binom{L-z}{M} = \binom{L+Q+1}{M+N+1} = \binom{n-1}{i-1}. \qquad (3)\end{aligned}$$

Note that in order to apply (R1) we need to verify $L, M \geq 0$ and $N \geq Q \geq 0$. We can assume $k \leq n/2$, since for $k > n/2$, there exist a trivial $(1/2)$-competitive algorithm. Therefore, we have $L = n-r-j-1 \geq n-k-(k-1)-1 = n-2k \geq 0$. Further, $i \geq t+j$, thus $i-j \geq t \geq k+1 \geq r+1$ which implies $M \geq 0$. The condition $N \geq Q \geq 0$ holds trivially. By inserting Equation (3) into Equation (2), we obtain the quotient of binomial coefficients $\binom{n-1}{i-1}/\binom{n-r-j-1}{i-j-r-1}$. From (R3) we get

$$\binom{n-1}{i-1}\Big/\binom{n-1-(r+j)}{i-1-(r+j)} = \binom{n-1}{r+j}\Big/\binom{i-1}{r+j} = \frac{(n-1)^{\underline{r+j}}}{(i-1)^{\underline{r+j}}}.$$

Recall $\beta = 1/n^{\underline{j+r+1}}$, thus $(n-1)^{\underline{r+j}}\cdot\beta = 1/n$. Together with Equation (2) we get

$$\mathbf{Pr}[E_j] = \beta\kappa\tau\cdot(n-1)^{\underline{r+j}}\sum_{i=t+j}^{n}\binom{i-t}{j}\frac{1}{(i-1)^{\underline{r+j}}} = \frac{\kappa\tau}{n}\sum_{i=t+j}^{n}\binom{i-t}{j}\frac{1}{(i-1)^{\underline{r+j}}}, \qquad (4)$$

which concludes the proof. ◀

Lemma 3.2 provides the exact probability that a dominating item is accepted as $(j+1)$-th item. However, it is more meaningful to consider the asymptotic setting where $n \to \infty$. Here, we assume $t-1 = cn$ for some constant $c \in (0,1)$. For this setting, we obtain the following lemma.

▶ **Lemma 3.3.** *Let E_j be defined as in Lemma 3.2. In the asymptotic setting described above,*

(A) *For $r = 1$ it holds that $\mathbf{Pr}[E_j] = c\left(\ln\frac{1}{c} + \sum_{\ell=1}^{j}\beta_\ell\frac{c^\ell-1}{\ell}\right)$, where $\beta_\ell = (-1)^{\ell+1}\binom{j}{\ell}$ for $\ell \in [j]$.*

(B) *For $r \geq 2$ it holds that $\mathbf{Pr}[E_j] = \frac{c}{r-1} - \frac{c^r(1-c)^j}{r-1}\sum_{\ell=0}^{j}\alpha_\ell\left(\frac{c}{1-c}\right)^\ell$, where $\alpha_\ell = \binom{j+r-1}{\ell+r-1}$ for $\ell \in [0..j]$.*

The proof of Lemma 3.3 relies on a sequence of technical lemmas and is given in Appendix A.

▶ Remark. As described in Section 2.1, SINGLE-REF generalizes the optimal strategy for the secretary problem ($k = 1$). Note that the combinatorial analysis from Lemma 3.2 as well as the asymptotic bound from Lemma 3.3 give exactly the respective terms from the secretary problem. To see this, we set $r = 1$ and consider the probability that the dominating item v_1 is accepted as first item. By Lemma 3.2 (with $j = 0$), the success probability is $\frac{t-1}{n}\sum_{i=t}^{n}\frac{1}{i-1}$. Moreover, Lemma 3.3(A) provides the asymptotic bound of $c\ln(1/c)$ for this case.

3.2 Non-Dominating Items

It remains to consider the acceptance probabilities of the non-dominating items $v_{r+1}, \ldots, v_k$. Fortunately, there exist some interesting connections to the probabilities for dominating items.

▶ **Lemma 3.4.** *Let $i \in [1..k-r]$ and $j \in [1..i]$. For the non-dominating item v_{r+i} it holds that $\mathbf{Pr}[v_{r+i}$ is j-th accept$] = \mathbf{Pr}[v_{r+i}$ is $(i+1)$-th accept$]$.*

Proof. First we argue that there are in total at least $i + 1$ accepts if v_{r+i} is accepted. Assuming that v_{r+i} is accepted, we have $s_r < v_{r+i}$. Let $\mathcal{S}$ be the set of elements which the algorithm may accept, i.e. $\mathcal{S} = \{v_1, \ldots, v_{r+i}\}$. Since s_r is the r-th best element in the sampling, at most $r - 1$ elements from $\mathcal{S}$ can be part of the sampling and thus at least $r + i - (r - 1) = i + 1$ elements from $\mathcal{S}$, including v_{r+i}, are accepted.

As described in Section 2.2, we construct a bijective function $f\colon P \to Q$ where P (resp. Q) is the set of permutations where v_{r+i} is the j-th (resp. $(i+1)$-th) accept. For each input sequence $\pi \in P$, let $a_1, \ldots, a_{i+1}$ with $a_j = v_{r+i}$ denote the first $i + 1$ accepts. The function f swaps the positions of $a_1, \ldots, a_{i+1}$ in a cyclic shift, such that $a_j = v_{r+i}$ is at the former position of a_{i+1}. In other words, the relative order of the first $i + 1$ accepted elements in $f(\pi)$ is changed in a way that v_{r+i} is the $(i+1)$-th accept in $f(\pi)$. Note that the cyclic shift can be reversed, thus f is bijective. ◀

While Lemma 3.4 relates the acceptance probabilities of a single non-dominating item, the claim of Lemma 3.5 is in a way orthogonal by relating probabilities of non-dominating items to those for dominating items.

▶ **Lemma 3.5.** *Let $i \in [1..k-r]$ and $j \in [1..k-i]$. For the non-dominating item v_{r+i} and any dominating item v^+ it holds that $\mathbf{Pr}[v_{r+i}$ is $(i+j)$-th accept$] = \mathbf{Pr}[v^+$ is $(i+j)$-th accept$]$.*

Proof. Let P be the set of permutations where v_{r+i} is the $(i+j)$-th accept and let Q contain those where v^+ is the $(i+j)$-th accept. We prove the claim by defining a bijective function $f\colon P \to Q$. Let f be the function that swaps v_{r+i} with v^+ in the input sequence.

Consider any input sequence $\pi \in P$. As v_{r+i} is accepted, $s_r < v_{r+i}$. We can argue that in $f(\pi)$ element s_r is still the r-th best element of the sampling: This holds clearly if no item is moved out of or into the sampling. Otherwise, f moves v_{r+i} into the sampling and v^+ outside. But since $s_r < v_{r+i} < v^+$, this does not change the role of s_r as the r-th best sampling element. Thus f is injective.

To prove that f is surjective, let $\pi' \in Q$ be any input sequence where v^+ is the $(i+j)$-th accept. We next consider the rank z of $s_r = v_z$. As there must be sampling elements $s_1, \ldots, s_{r-1}$ and accepted elements $a_1, \ldots, a_{i+j-1}, v^+$ of rank smaller than z, we have $z > (r-1) + (i+j-1) + 1 \geq r + i$. Hence, $s_r < v_{r+i}$. The inverse function of f consists in swapping back v^+ with v_{r+i}. For the same reason as above, this maintains s_r. As $s_r < v_{r+i}$, element v_{r+i} gets accepted, thus $f^{-1}(\pi') \in P$. ◀

Using the previous results for dominating and non-dominating items we are now ready to state the main result of this section, namely the competitive ratio of SINGLE-REF. Due to the complex expressions from Lemma 3.3 we give numerical results for small values of k.

▶ **Theorem 3.6.** *In the asymptotic setting of $n \to \infty$ and assuming that $t - 1 = cn$ for a constant $c \in (0,1)$,* SINGLE-REF *achieves the competitive ratios given in Table 1.*

Proof. For an item v_i, let $p_i^{(j)}$ be the probability that v_i is the j-th accept (with $1 \leq j \leq k$). The total acceptance probability of v_i is denoted by $p_i = \sum_{j=1}^{k} p_i^{(j)}$. According to Lemma 3.2, each dominating item has the same acceptance probability for a fixed acceptance position. Therefore, in the following we simply write p_1 (resp. $p_1^{(j)}$) for the acceptance probability of any dominating item.

By Lemma 2.1 the competitive ratio can be obtained by summing over the acceptance probabilities of all items divided by k. Clearly, $\sum_{i=1}^{r} p_i = rp_1$. Now consider any non-dominating item v_{r+i}. According to Lemmas 3.4 and 3.5, p_{r+i} can be related to respective probabilities $p_1^{(j)}$: It holds that $p_{r+i}^{(j)} = p_1^{(z)}$ with $z = \max\{j, i+1\}$. Therefore $p_{r+i} = \sum_{j=1}^{k} p_{r+i}^{(j)} = \sum_{j=1}^{i} p_1^{(i+1)} + \sum_{j=i+1}^{k} p_1^{(j)} = ip_1^{(i+1)} + \sum_{j=i+1}^{k} p_1^{(j)}$. Hence, we obtain the competitive ratio

$$\frac{1}{k}\sum_{i=1}^{k} p_i = \frac{1}{k}\left(rp_1 + \sum_{i=1}^{k-r}\left(ip_1^{(i+1)} + \sum_{j=i+1}^{k} p_1^{(j)}\right)\right) \tag{5}$$

with $p_1^{(j)} = \mathbf{Pr}\left[E_{j-1}\right]$ for the event E_j considered in Lemmas 3.2 and 3.3. To evaluate the performance of our algorithm, we maximized Equation (5) over the parameters r and c using a computer algebra system. This yields the competitive ratios shown in Table 1 (p. 3). ◀

For completeness, we evaluated the competitive ratio of SINGLE-REF in the interval $k \in [1..100]$ using the optimization procedure mentioned in the previous proof. Figure 2 shows the performance of SINGLE-REF in comparison with Kleinberg's result [21]; our algorithm reaches competitive ratios of up to 0.75 and outperforms the algorithm from [21] on this interval. In Appendix A, we provide the full list of optimal parameters for $k \in [1..100]$ (see Table 3, p. 19).

4 Analysis of OPTIMISTIC for $k = 2$

In this section we sketch the analysis of OPTIMISTIC for $k = 2$. Due to space constraints, for some proofs we refer to the full version of this paper. Let $\mathcal{A}_2$ denote OPTIMISTIC algorithm with $k = 2$ in the following. As implied by Lemma 2.1 the competitive ratio is determined by p_1 and p_2, the probabilities that $\mathcal{A}_2$ accepts v_1 and v_2, respectively. To find these probabilities, we make use of the relation between probabilities and sets (see Section 2.2). Let P_i be the set of permutations in which $\mathcal{A}_2$ accepts v_i.

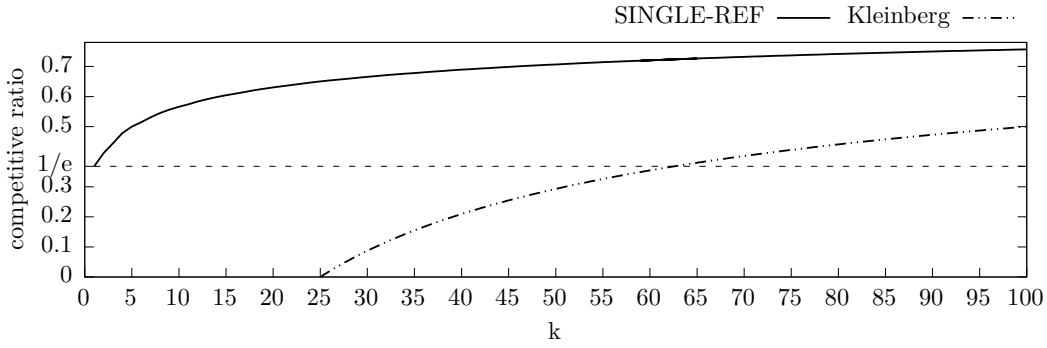


Figure 2 Comparison of our algorithm SINGLE-REF and the algorithm by Kleinberg [21], for $k \in [1..100]$. The parameters r and c for SINGLE-REF are chosen optimally.

Probability p_2. In the next lemma, we show a surprising relation between OPTIMISTIC for $k = 2$ and the algorithm for the classical (1-)secretary problem (see Section 2.1). The proof of Lemma 4.1 uses a sophistically tailored bijection between two respective sets of permutations. We sketch the proof method here and give the entire proof in the full version of this paper.

▶ **Lemma 4.1.** *Let $\mathcal{A}_1$ be the algorithm for the classical secretary problem. Assuming that both algorithms $\mathcal{A}_1$, $\mathcal{A}_2$ are parameterized with the same t, we have that $p_2 = \mathbf{Pr}[\mathcal{A}_2 \text{ accepts } v_2] = \mathbf{Pr}[\mathcal{A}_1 \text{ accepts } v_1]$.*

Sketch of proof. Equivalently, we prove that the corresponding complementary events happen with the same probability. For this purpose, we define for each permutation π where $\mathcal{A}_2$ does not accept v_2 a unique permutation $f(\pi)$ where $\mathcal{A}_1$ does not accept v_1. Different situations where $\mathcal{A}_2$ does not accept v_2 lead to a total number of five cases. If v_2 is in the sampling of π, we define $f(\pi)$ such that the positions of v_1 and v_2 are swapped. Then, $\mathcal{A}_1$ clearly does not accept v_1 in $f(\pi)$. Another case is when v_2 comes behind two accepted elements a_1, a_2 in π and $v_1 = a_1$ is the first accept. Note that since a_2 is accepted, $a_2 > s_1$. In this case, $f(\pi)$ can be defined by swapping the positions of both accepts v_1 and a_2. Recall that $\mathcal{A}_1$ accepts the first item better than s_1 following the sampling phase which is a_2 in $f(\pi)$, thus v_1 is not selected.

In the full proof, we consider all five cases according to π. In each case it is enough to define f such that the positions of at most three elements are swapped. Finally, we have to argue that the function f is indeed bijective. ◀

Probability p_1. In this part, we argue that $p_1 = p_2 + \delta$ holds for some $\delta > 0$. To obtain δ, we again consider cardinalities of sets instead of probabilities. First, we observe that P_2 can be related to a set $P_1' \subset P_1$ such that P_2 and P_1' have equal size.

▶ **Lemma 4.2.** *Let $P_1' = \{\pi \in P_1 \mid \text{pos}_\pi(v_2) < t \Rightarrow \mathcal{A}_2 \text{ accepts } v_1 \text{ as first item}\}$. It holds that $|P_1'| = |P_2|$.*

Proof. Let $f\colon P_2 \to P_1'$ be the function that swaps v_1 with v_2 in the given sequence. We first have to argue that in fact $f\colon P_2 \to P_1'$, therefore let $\pi \in P_2$ be given. Then, v_1 gets accepted by $\mathcal{A}_2$ in $f(\pi)$ at the position $\text{pos}_{f(\pi)}(v_1) = \text{pos}_\pi(v_2)$, as v_1 is an item of higher value. So far we have $f(\pi) \in P_1$. If $\text{pos}_{f(\pi)}(v_2) \geq t$, there is nothing to show. Assuming that $\text{pos}_{f(\pi)}(v_2) < t$, it follows $\text{pos}_\pi(v_1) < t$, i.e. v_1 was the best element in the sampling of π. Since no item (particularly not v_2) can beat v_1, but v_2 was accepted by $\mathcal{A}_2$ in π, we get that v_2 was the first accept in π. Hence v_1 is the first accept in $f(\pi)$.

Clearly, f is injective. For surjectivity, let $\pi' \in P_1'$ and let π the permutation obtained from π' by swapping (back) v_1 with v_2. If $\text{pos}_{\pi'}(v_2) < t$, by definition of P_1' we know that v_1 is the first accept in π', implying that no item before $\text{pos}_{\pi'}(v_1) = \text{pos}_{\pi}(v_2)$ is chosen by $\mathcal{A}_2$. In the case $\text{pos}_{\pi'}(v_2) \geq t$, since $\text{pos}_{\pi'}(v_1) \geq t$, the smallest rank in the sampling of π' is 3 or greater. Therefore, v_2 gets accepted if not more than one item before v_2 gets accepted. This is the case in π, as $\text{pos}_{\pi}(v_2) = \text{pos}_{\pi'}(v_1)$. ◀

Since $|P_1| = |P_1'| + |P_1 \setminus P_1'| = |P_2| + |P_1 \setminus P_1'|$, we therefore get $\delta = |P_1 \setminus P_1'|/n!$, i.e., δ is the probability that a random permutation is in the set $|P_1 \setminus P_1'|$. This probability is considered in Lemma 4.3.

▶ **Lemma 4.3.** *Let $\delta = \mathbf{Pr}[\pi \in P_1 \setminus P_1']$ where π is drawn uniformly from the set of all permutations and P_1' is defined like in Lemma 4.2. It holds that $\delta = \frac{t-1}{n}\frac{t-2}{n-1}\sum_{i=t}^{n-1}\frac{n-i}{(i-2)(i-1)}$.*

The proof of Lemma 4.3 relies on a counting argument similar to the proof of Lemma 3.2. We prove Lemma 4.3 in the full version of this paper.

Competitive ratio. From Lemmas 4.1 and 4.3, we know the exact probabilities p_2 and p_1. For particular n, the term $(p_1 + p_2)/2$ can be optimized over t to find the optimal sampling size. In the following theorem we consider the asymptotic setting $n \to \infty$. Here, we assume that the sampling size is a constant fraction of the input size, i.e., $t - 1 = cn$ for some constant $c \in (0, 1)$.

▶ **Theorem 4.4.** *For $k = 2$, the algorithm* OPTIMISTIC *is 0.4168-competitive in the limit $n \to \infty$ and assuming that the sampling size is $t - 1 = cn$ for $c = 0.3521$.*

Proof. According to Lemma 4.1, p_2 is the probability that the classical secretary algorithm accepts the best item, i.e., $p_2 = \frac{t-1}{n}\sum_{i=t}^{n}\frac{1}{i-1}$. This term approaches $c\ln(1/c)$ asymptotically. From Lemma 4.3 we know $p_1 = p_2 + \delta$, where $\delta = \frac{t-1}{n}\frac{t-2}{n-1}\sum_{i=t}^{n-1}\frac{n-i}{(i-2)(i-1)}$. For $n \to \infty$, the sum $\sum_{i=t}^{n-1}\frac{n-i}{(i-2)(i-1)}$ is bounded from above and below by $\frac{1}{c} - \ln\frac{1}{c} - 1$. This can be seen by bounding the sum by two corresponding integrals. Further, $\lim_{n\to\infty}\frac{t-1}{n}\frac{t-2}{n-1} = c^2$. Therefore, $\delta = c^2\left(\frac{1}{c} - \ln\frac{1}{c} - 1\right)$ for large n. According to Lemma 2.1, $\mathcal{A}_2$ is $\alpha(c)$-competitive with

$$\alpha(c) = \frac{1}{2}(p_1 + p_2) = \frac{1}{2}(p_2 + \delta + p_2) = c\ln\frac{1}{c} + \frac{c^2}{2}\left(\frac{1}{c} - \ln\frac{1}{c} - 1\right).$$

Setting $c = 1/e$, we obtain a competitive ratio of $\alpha(1/e) = \frac{3e-2}{2e^2} \approx 0.4164$. However, the optimal choice for c is around $c^* = 0.3521 < 1/e$, improving the competitive ratio slightly to $\alpha(c^*) \approx 0.4168$. ◀

5 Conclusion and Future Work

We investigated two algorithms for the k-secretary problem with a focus on small values for $k \geq 2$. Aside from a tight analysis of the OPTIMISTIC algorithm [3] for $k = 2$, we introduced and analyzed the algorithm SINGLE-REF. For any value of k, the competitive ratio of SINGLE-REF can be obtained by numerical optimization.

We see various directions of future work. For SINGLE-REF, it remains to find the right dependency between the parameters r, c, and k in general and to find a closed formula for the competitive ratio for any value of k. OPTIMISTIC seems a promising and elegant algorithm, however no tight analysis for general $k \geq 3$ is known so far. For $k = 2$, we identified a key property in Lemma 4.1. Similar properties may hold in the general case. Lastly, to the best of our knowledge, no hardness results for the k-secretary problem are known (apart from the cases $k \leq 2$).

References

1. S. Agrawal, Z. Wang, and Y. Ye. A Dynamic Near-Optimal Algorithm for Online Linear Programming. *Operations Research*, 62(4):876–890, 2014.
2. M. Ajtai, N. Megiddo, and O. Waarts. Improved algorithms and analysis for secretary problems and generalizations. *SIAM Journal on Discrete Mathematics*, 14(1):1–27, 2001.
3. M. Babaioff, N. Immorlica, D. Kempe, and R. Kleinberg. A Knapsack Secretary Problem with Applications. In *Proc. 10th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems and 11th International Workshop on Randomization and Computation (APPROX/RANDOM)*, pages 16–28, 2007.
4. M. Babaioff, N. Immorlica, D. Kempe, and R. Kleinberg. Online auctions and generalized secretary problems. *SIGecom Exchanges*, 7(2), 2008.
5. M. Babaioff, N. Immorlica, D. Kempe, and R. Kleinberg. Matroid Secretary Problems. *Journal of the ACM (JACM)*, 65(6):35:1–35:26, 2018.
6. M. Babaioff, N. Immorlica, and R. Kleinberg. Matroids, secretary problems, and online mechanisms. In *Proc. 18th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 434–443, 2007.
7. B. Bahmani, A. Mehta, and R. Motwani. A 1.43-Competitive Online Graph Edge Coloring Algorithm in the Random Order Arrival Model. In *Proc. 21st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 31–39, 2010.
8. N. Buchbinder, K. Jain, and M. Singh. Secretary Problems via Linear Programming. *Mathematics of Operations Research*, 39(1):190–206, 2014.
9. T.-H. H. Chan, F. Chen, and S. H.-C. Jiang. Revealing Optimal Thresholds for Generalized Secretary Problem via Continuous LP: Impacts on Online K-Item Auction and Bipartite K-Matching with Random Arrival Order. In *Proc. 26th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1169–1188, 2015.
10. E. B Dynkin. The optimum choice of the instant for stopping a Markov process. *Soviet Mathematics*, 4:627–629, 1963.
11. M. Feldman, O. Svensson, and R. Zenklusen. A Simple $O(\log \log(\text{rank}))$-Competitive Algorithm for the Matroid Secretary Problem. In *Proc. 26th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1189–1201, 2015.
12. M. Feldman, O. Svensson, and R. Zenklusen. A Framework for the Secretary Problem on the Intersection of Matroids. In *Proc. 29th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 735–752, 2018.
13. T. S. Ferguson. Who Solved the Secretary Problem? *Statistical Science*, 4(3):282–289, 1989.
14. P.R. Freeman. The secretary problem and its extensions: A review. *International Statistical Review/Revue Internationale de Statistique*, pages 189–206, 1983.
15. O. Göbel, T. Kesselheim, and A. Tönnis. Online Appointment Scheduling in the Random Order Model. In *Proc. 23rd Annual European Symposium on Algorithms (ESA)*, pages 680–692, 2015.
16. R. L. Graham, D. E. Knuth, and O. Patashnik. *Concrete mathematics - a foundation for computer science (2. ed.)*. Addison-Wesley, 1994.
17. M. Hoefer and B. Kodric. Combinatorial Secretary Problems with Ordinal Information. In *44th International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 133:1–133:14, 2017.
18. C. Kenyon. Best-Fit Bin-Packing with Random Order. In *Proc. 7th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 359–364, 1996.
19. T. Kesselheim, K. Radke, A. Tönnis, and B. Vöcking. Primal beats dual on online packing LPs in the random-order model. In *Proc. 46th Annual ACM Symposium on Theory of Computing (STOC)*, pages 303–312, 2014.
20. T. Kesselheim and A. Tönnis. Submodular Secretary Problems: Cardinality, Matching, and Linear Constraints. In *Proc. 20th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems and 21st International Workshop on Randomization and Computation (APPROX/RANDOM)*, pages 16:1–16:22, 2017.

21 R. D. Kleinberg. A multiple-choice secretary algorithm with applications to online auctions. In *Proc. 16th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 630–631, 2005.

22 O. Lachish. O(log log Rank) Competitive Ratio for the Matroid Secretary Problem. In *Proc. 55th IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, pages 326–335, 2014.

23 D. V Lindley. Dynamic programming and decision theory. *Applied Statistics*, pages 39–51, 1961.

24 M. Mahdian and Q. Yan. Online bipartite matching with random arrivals: an approach based on strongly factor-revealing LPs. In *Proc. 43rd ACM Symposium on Theory of Computing (STOC)*, pages 597–606, 2011.

A Technical Proofs for SINGLE-REF

In several lemmas we need to find closed expressions for sums over values of a certain function. If the function is monotone, such sums can be bounded by corresponding integrals:

▶ **Fact A.1.** *Let $f\colon \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ and $a, b \in \mathbb{N}$.*

(A) *If f is monotonically decreasing, then $\int_a^{b+1} f(i)\ \mathrm{d}i \ \leq \sum_{i=a}^{b} f(i) \leq \int_{a-1}^{b} f(i)\ \mathrm{d}i$.*

(B) *If f is monotonically increasing, then $\int_{a-1}^{b} f(i)\ \mathrm{d}i \ \leq \sum_{i=a}^{b} f(i) \leq \int_a^{b+1} f(i)\ \mathrm{d}i$.*

In Lemma 3.3 we consider the acceptance probabilities of dominating items in the asymptotic setting $n \to \infty$ with $t - 1 = cn$ for $c \in (0, 1)$. We can assume further that $j, r \leq k = o(n)$. In the following, we prove Lemma 3.3 using some technical lemmas, stated and proven below the main proof.

Proof of Lemma 3.3. We first consider the sum $S := \sum_{i=t+j}^{n} \binom{i-t}{j} \frac{1}{(i-1)^{\underline{r+j}}}$ from Equation (4) and obtain the following lower bound:

$$S = \sum_{i=t+j}^{n} \binom{i-t}{j} \frac{1}{(i-1)^{\underline{r+j}}} = \frac{1}{j!} \sum_{i=t+j}^{n} \frac{(i-t)^{\underline{j}}}{(i-1)^{\underline{r+j}}} \geq \frac{1}{j!} \sum_{i=t+j}^{n} \frac{(i-t-j+1)^j}{(i-1)^{r+j}}$$
$$= \frac{1}{j!} \sum_{i=1}^{n-t-j+1} \frac{i^j}{(i+t+j-2)^{r+j}}\,.$$

Let $f(i) = i^j/(i+y)^{r+j}$ for $y = t + j - 2$. Note that y can be seen as a constant independent from i. Let $m = n - t - j + 1$, now the above inequality reads as $S \geq (1/j!) \sum_{i=1}^{m} f(i)$. In the following we investigate the function f.

Unfortunately, f is in general not monotone, hence we can not apply Fact A.1A or Fact A.1B directly in order to bound the sum by an integral. However, we can split the sum into two monotone parts. Let d be defined like in Lemma A.2 (following this proof). Now we can apply Fact A.1 as follows:

$$\sum_{i=1}^{m} f(i) = \sum_{i=1}^{d} f(i) + \sum_{i=d+1}^{m} f(i) \geq \int_0^d f(i)\ \mathrm{d}i \ + \int_{d+1}^{m+1} f(i)\ \mathrm{d}i$$
$$= \int_0^{m+1} f(i)\ \mathrm{d}i \ - \int_d^{d+1} f(i)\ \mathrm{d}i\,. \tag{6}$$

Finding the indefinite integral $\int f(i)\,\mathrm{d}i$ turns out to be a technical task and is therefore moved to separate lemmas (see Lemmas A.3 and A.4). If $F(i)$ is a function with $F'(i) = f(i)$, we have for κ, τ defined like in Equation (4)

$$\mathbf{Pr}\left[E_j\right] = \frac{\kappa\tau}{n} S \geq \frac{\kappa\tau}{nj!}\left(F(m+1) - F(0) - F(d+1) + F(d)\right) . \tag{7}$$

In the remainder of the proof we consider the two cases $r = 1$ and $r \geq 2$ separately.

Case A: $r = 1$. Let $F(i)$ and β_ℓ be defined like in Lemma A.3. In Equation (7), the factor $\frac{\kappa\tau}{nj!}$ resolves to c as $\kappa = (j+r-1)^{\underline{j}} = j^{\underline{j}} = j!$ and $\tau = (t-1)^{\underline{r}} = (t-1)^{\underline{1}} = t-1$. Further it holds that

$$\begin{aligned}\lim_{n\to\infty} F(m+1) &= \lim_{n\to\infty}\left(\ln((m+1)+y) + \sum_{\ell=1}^{j}\beta_\ell \frac{y^\ell}{\ell((m+1)+y)^\ell}\right)\\ &= \lim_{n\to\infty}\left(\ln n + \sum_{\ell=1}^{j}\beta_\ell\frac{(t+j-2)^\ell}{\ell n^\ell}\right) = \lim_{n\to\infty}\left(\ln n + \sum_{\ell=1}^{j}\beta_\ell\frac{c^\ell}{\ell}\right)\end{aligned}$$

and moreover

$$\begin{aligned}\lim_{n\to\infty} F(0) &= \lim_{n\to\infty}\left(\ln y + \sum_{\ell=1}^{j}\beta_\ell\frac{y^\ell}{\ell y^\ell}\right) = \lim_{n\to\infty}\left(\ln(t+j-2) + \sum_{\ell=1}^{j}\beta_\ell\frac{1}{\ell}\right)\\ &= \lim_{n\to\infty}\left(\ln t + \sum_{\ell=1}^{j}\beta_\ell\frac{1}{\ell}\right) .\end{aligned}$$

Hence, $\lim_{n\to\infty}\left(F(m+1) - F(0)\right) = \ln\frac{1}{c} + \sum_{\ell=1}^{j}\beta_\ell\frac{c^\ell-1}{\ell}$. It remains to consider $F(d) - F(d+1)$ in the limit of $n\to\infty$. It holds that

$$\begin{aligned}F(d) - F(d+1) &= \ln(d+y) + \sum_{\ell=1}^{j}\beta_\ell\frac{y^\ell}{\ell(d+y)^\ell} - \ln(d+1+y) - \sum_{\ell=1}^{j}\beta_\ell\frac{y^\ell}{\ell(d+1+y)^\ell}\\ &= \ln\left(\frac{d+y}{d+1+y}\right) + \sum_{\ell=1}^{j}\frac{\beta_\ell}{\ell}\left(\left(\frac{y}{d+y}\right)^\ell - \left(\frac{y}{d+1+y}\right)^\ell\right)\end{aligned}$$

and since $y = t+j-2 = \Theta(n)$ and $d = (j/r)y = \Theta(y)$, we get that $\lim_{n\to\infty}\left(F(d) - F(d+1)\right) = 0$.

Case B: $r \geq 2$. In this case let $F(i)$ and α_ℓ be defined according to Lemma A.4. Further, let $G(i) = -\alpha_0(r-1)F(i)$. Using Equation (7) we obtain

$$\begin{aligned}&= \frac{\kappa\tau}{nj!\alpha_0(r-1)}\left(G(0) - G(m+1) + G(d+1) - G(d)\right)\\ &= \frac{\tau}{n(r-1)}\left(G(0) - G(m+1) + G(d+1) - G(d)\right) ,\end{aligned} \tag{8}$$

where the last equality follows from the definition of $\alpha_0 = \binom{j+r-1}{r-1} = \kappa/j!$. We first notice

$$\lim_{n\to\infty}\frac{\tau}{n(r-1)} = \frac{1}{r-1}\lim_{n\to\infty}\frac{(t-1)^{\underline{r}}}{n} = \frac{1}{r-1}\lim_{n\to\infty}\frac{(t-1)^r}{n} = \frac{1}{r-1}c^r\lim_{n\to\infty} n^{r-1} .$$

Further it holds that

$$G(m+1) = \frac{\sum_{\ell=0}^{j}\alpha_\ell(m+1)^{j-\ell}(t+j-2)^\ell}{(m+1+t+j-2)^{r+j-1}} = \frac{\sum_{\ell=0}^{j}\alpha_\ell(m+1)^{j-\ell}(t+j-2)^\ell}{n^{r+j-1}} .$$

Note that $\lim_{n\to\infty}(m+1) = \lim_{n\to\infty}(n-(t-1)) = \lim_{n\to\infty}(n-cn) = \lim_{n\to\infty}(1-c)n$ and similarly $\lim_{n\to\infty}(t+j-2) = \lim_{n\to\infty}(t-1) = \lim_{n\to\infty} cn$. Hence we get

$$\lim_{n\to\infty} G(m+1) = \lim_{n\to\infty} \frac{\sum_{\ell=0}^{j} \alpha_\ell (1-c)^{j-\ell} n^{j-\ell} c^\ell n^\ell}{n^{r+j-1}} = \lim_{n\to\infty} \frac{\sum_{\ell=0}^{j} \alpha_\ell (1-c)^{j-\ell} c^\ell}{n^{r-1}}\,.$$

For the term $G(0)$ we obtain

$$G(0) = \frac{\sum_{\ell=0}^{j} \alpha_\ell 0^{j-\ell} y^\ell}{y^{r+j-1}} = \frac{\alpha_j y^j}{y^{r+j-1}} = \frac{1}{y^{r-1}}$$

and thus $\lim_{n\to\infty} G(0) = \lim_{n\to\infty} \frac{1}{y^{r-1}} = \lim_{n\to\infty} \frac{1}{(t-1)^{r-1}} = \frac{1}{c^{r-1}} \lim_{n\to\infty} \frac{1}{n^{r-1}}$.
In Equation (8) it remains to consider $G(d+1) - G(d)$. Similarly to case A we can show that this term approaches 0 for $n \to \infty$:

$$\begin{aligned} G(d+1) - G(d) &= \frac{\sum_{\ell=0}^{j} \alpha_\ell (d+1)^{j-\ell} y^\ell}{(d+1+y)^{r+j-1}} - \frac{\sum_{\ell=0}^{j} \alpha_\ell d^{j-\ell} y^\ell}{(d+y)^{r+j-1}} \\ &\le \frac{\sum_{\ell=0}^{j} \alpha_\ell y^\ell \left((d+1)^{j-\ell} - d^{j-\ell}\right)}{(d+y)^{r+j-1}} \end{aligned}$$

where the numerator approaches 0 since $d = \Theta(y) = \Theta(n)$. Using Equation (8) and all limits stated above, we get finally

$$\begin{aligned} \lim_{n\to\infty} \mathbf{Pr}\left[E_j\right] &= \lim_{n\to\infty} \frac{1}{r-1} c^r n^{r-1} \left(\frac{1}{c^{r-1}} \frac{1}{n^{r-1}} - \frac{\sum_{\ell=0}^{j} \alpha_\ell (1-c)^{j-\ell} c^\ell}{n^{r-1}} \right) \\ &= \frac{1}{r-1} \left(c - \sum_{\ell=0}^{j} \alpha_\ell c^{r+\ell} (1-c)^{j-\ell} \right) \\ &= \frac{c}{r-1} - \frac{c^r (1-c)^j}{r-1} \sum_{\ell=0}^{j} \alpha_\ell \left(\frac{c}{1-c} \right)^\ell . \end{aligned}$$

This concludes the proof. ◀

▶ **Lemma A.2.** *Let $f\colon \mathbb{R} \to \mathbb{R}$ with $f(i) = i^j/(i+y)^{r+j}$ and $j \ge 0$, $r \ge 1$, and $y > 0$ does not depend on i. The function f is monotonically increasing for $i \le d$ and monotonically decreasing for $i > d$ where $d = (jy)/r$.*

Proof. Let $g(i) = i^j$ and $h(i) = (i+y)^{r+j}$. We consider the first derivative $f'(i) = \frac{g'(i)h(i) - g(i)h'(i)}{h(i)^2}$. Since $h(i)^2$ is nonnegative, f grows monotonically if

$$g'(i)h(i) \ge g(i)h'(i) \;\Leftrightarrow\; ji^{j-1}(i+y)^{r+j} \ge i^j (r+j)(i+y)^{r+j-1} \;\Leftrightarrow\; j(i+y) \ge i(r+j)\,.$$

It is easy to see that the last inequality is equivalent to $i \le \frac{jy}{r} = d$. ◀

▶ **Lemma A.3.** *Let $f\colon \mathbb{R} \to \mathbb{R}$ with $f(i) = i^j/(i+y)^{r+j}$ and $r = 1$, $j \ge 0$, and $y > 0$ does not depend on i. The following function F fulfills $F'(i) = f(i)$:*

$$F(i) = \ln(i+y) + \sum_{\ell=1}^{j} \beta_\ell \frac{y^\ell}{\ell (i+y)^\ell}$$

where $\beta_\ell = (-1)^{\ell+1} \binom{j}{\ell}$ for $1 \le \ell \le j$.

Proof. We need to show $F'(i) = f(i)$ and observe first that

$$F'(i) = \frac{1}{i+y} + \sum_{\ell=1}^{j} \beta_\ell \frac{-\ell y^\ell}{\ell(i+y)^{\ell+1}} = \frac{1}{i+y} + \sum_{\ell=1}^{j} \beta_\ell y^\ell \frac{-(i+y)^{j-\ell}}{(i+y)^{j+1}}$$
$$= \frac{1}{(i+y)^{j+1}} \left((i+y)^j + \sum_{\ell=1}^{j} \beta_\ell y^\ell (-(i+y)^{j-\ell}) \right)$$

and since $\beta_0 = (-1)^{0+1}\binom{j}{0} = -1$ we get further

$$F'(i) = \frac{1}{(i+y)^{j+1}} \sum_{\ell=0}^{j} \beta_\ell y^\ell (-(i+y)^{j-\ell}) = \frac{1}{(i+y)^{j+1}} \sum_{\ell=0}^{j} (-1)^{\ell+2} \binom{j}{\ell} y^\ell (i+y)^{j-\ell}.$$

Finally, note that $(-1)^{\ell+2} y^\ell = (-y)^\ell$, thus by the binomial theorem the last sum evaluates to $((i+y)+(-y))^j = i^j$ which concludes the proof. ◀

▶ **Lemma A.4.** *Let $f\colon \mathbb{R} \to \mathbb{R}$ with $f(i) = i^j/(i+y)^{r+j}$ and $j \geq 0$, $r \geq 2$, and $y > 0$ does not depend on i. The following function F fulfills $F'(i) = f(i)$:*

$$F(i) = -\frac{\sum_{\ell=0}^{j} \alpha_\ell i^{j-\ell} y^\ell}{\alpha_0 (r-1)(i+y)^{r+j-1}},$$

where $\alpha_\ell = \binom{j+r-1}{\ell+r-1}$ for $0 \leq \ell \leq j$.

Proof. Let $G(i)$ and $H(i)$ be the numerator and denominator of $F(i)$. It holds that $G'(i) = -\sum_{\ell=0}^{j} \alpha_\ell (j-\ell) i^{j-\ell-1} y^\ell$ and $H'(i) = \alpha_0 (r-1)(r+j-1)(i+y)^{r+j-2} = H(i) r(i)$ where $r(i) = \frac{r+j-1}{i+y}$. In order to prove the claim, we show

$$G'(i)(i+y) - G(i)(r+j-1) = i^j \alpha_0 (r-1) \tag{9}$$

since then we have

$$F'(i) = \frac{G'(i)H(i) - G(i)H'(i)}{H(i)^2} = \frac{G'(i) - G(i)r(i)}{H(i)} = \frac{G'(i) - G(i)r(i)}{\frac{\alpha_0(r-1)}{i+y}(i+y)^{r+j}}$$
$$= \frac{(i+y)(G'(i) - G(i)r(i))}{\alpha_0(r-1)(i+y)^{r+j}} = \frac{(i+y)G'(i) - (r+j-1)G(i)}{\alpha_0(r-1)(i+y)^{r+j}}$$

With Equation (9), the last term resolves to $= \frac{i^j \alpha_0 (r-1)}{\alpha_0 (r-1)(i+y)^{r+j}} = f(i)$. It remains to show Equation (9):

$$\begin{aligned}
&G'(i)(i+y) - G(i)(r+j-1) \\
&= -\left(\sum_{\ell=0}^{j} \alpha_\ell (j-\ell) i^{j-\ell-1} y^\ell \right)(i+y) + \left(\sum_{\ell=0}^{j} \alpha_\ell i^{j-\ell} y^\ell \right)(r+j-1) \\
&= -\left(\sum_{\ell=0}^{j} \alpha_\ell (j-\ell) i^{j-\ell} y^\ell \right) - \left(\sum_{\ell=0}^{j} \alpha_\ell (j-\ell) i^{j-\ell-1} y^{\ell+1} \right) \\
&\quad + \left(\sum_{\ell=0}^{j} \alpha_\ell i^{j-\ell} y^\ell \right)(r+j-1) \\
&= \left(\sum_{\ell=0}^{j} \alpha_\ell i^{j-\ell} y^\ell (r-1+\ell) \right) - \left(\sum_{\ell=0}^{j-1} \alpha_\ell (j-\ell) i^{j-\ell-1} y^{\ell+1} \right).
\end{aligned}$$

Note that the first sum contains all powers of i from i^0 to i^j, while the latter sum only powers from i^0 to i^{j-1}. Therefore, we can split up the part for i^j from the first sum and group equal powers of i to obtain

$$\alpha_0(r-1)i^j + \sum_{\ell=1}^{j} \left(\alpha_\ell(r-1+\ell) - \alpha_{\ell-1}(j-\ell+1)\right) i^{j-\ell}y^\ell \,.$$

The claim follows if we can show that the last sum evaluates to zero. This is true, since by definition of α_ℓ it holds that

$$\alpha_\ell(r-1+\ell) = \binom{j+r-1}{\ell+r-1}(r-1+\ell) = \frac{(j+r-1)!}{(\ell+r-1)!(j-\ell)!}(r-1+\ell)$$
$$= \frac{(j+r-1)!}{(\ell+r-2)!(j-\ell+1)!}\frac{(j-\ell+1)}{(j-\ell)!} = \binom{j+r-1}{(\ell-1)+r-1}(j-\ell+1) = \alpha_{\ell-1}(j-\ell+1)\,. \quad \blacktriangleleft$$

Table 3 Optimal parameters and corresponding competitive ratios of SINGLE-REF for $k \in [1..100]$. For readibility, the numeric values are truncated after the fourth decimal place.

k	r	c	competitive ratio
1	1	0.3678	0.3678
2	1	0.2545	0.4119
3	2	0.3475	0.4449
4	2	0.2928	0.4785
5	2	0.2525	0.4999
6	2	0.2217	0.5148
7	3	0.2800	0.5308
8	3	0.2549	0.5453
9	3	0.2338	0.5567
10	3	0.2159	0.5660
11	4	0.2570	0.5740
12	4	0.2410	0.5834
13	4	0.2267	0.5914
14	4	0.2140	0.5983
15	4	0.2026	0.6043
16	4	0.1924	0.6096
17	5	0.2231	0.6155
18	5	0.2133	0.6211
19	5	0.2042	0.6261
20	5	0.1959	0.6306
21	5	0.1882	0.6347
22	5	0.1811	0.6384
23	6	0.2054	0.6426
24	6	0.1985	0.6465
25	6	0.1919	0.6502
26	6	0.1858	0.6535
27	6	0.1800	0.6566
28	6	0.1746	0.6595
29	7	0.1947	0.6625
30	7	0.1893	0.6655
31	7	0.1842	0.6684
32	7	0.1793	0.6711
33	7	0.1747	0.6736
34	7	0.1703	0.6760
35	7	0.1662	0.6782
36	8	0.1830	0.6805
37	8	0.1788	0.6829
38	8	0.1748	0.6851
39	8	0.1710	0.6873
40	8	0.1673	0.6893
41	8	0.1638	0.6912
42	8	0.1605	0.6930
43	9	0.1750	0.6948
44	9	0.1716	0.6968
45	9	0.1683	0.6986
46	9	0.1651	0.7004
47	9	0.1621	0.7021
48	9	0.1592	0.7037
49	9	0.1563	0.7052
50	9	0.1536	0.7067
51	10	0.1662	0.7082
52	10	0.1635	0.7098
53	10	0.1608	0.7113
54	10	0.1582	0.7127
55	10	0.1557	0.7141
56	10	0.1532	0.7155
57	10	0.1509	0.7168
58	10	0.1486	0.7180
59	11	0.1597	0.7193
60	11	0.1574	0.7206
61	11	0.1551	0.7219
62	11	0.1529	0.7231
63	11	0.1508	0.7243
64	11	0.1487	0.7255
65	11	0.1467	0.7266
66	11	0.1447	0.7277
67	11	0.1428	0.7287
68	12	0.1527	0.7298
69	12	0.1508	0.7309
70	12	0.1489	0.7320
71	12	0.1470	0.7330
72	12	0.1452	0.7340
73	12	0.1434	0.7350
74	12	0.1417	0.7360
75	12	0.1400	0.7369
76	12	0.1384	0.7378
77	13	0.1473	0.7387
78	13	0.1456	0.7397
79	13	0.1440	0.7406
80	13	0.1424	0.7415
81	13	0.1408	0.7424
82	13	0.1393	0.7433
83	13	0.1378	0.7441
84	13	0.1363	0.7449
85	13	0.1349	0.7457
86	14	0.1429	0.7465
87	14	0.1415	0.7473
88	14	0.1400	0.7482
89	14	0.1386	0.7490
90	14	0.1372	0.7497
91	14	0.1359	0.7505
92	14	0.1346	0.7512
93	14	0.1333	0.7520
94	14	0.1320	0.7527
95	14	0.1307	0.7534
96	15	0.1381	0.7541
97	15	0.1368	0.7548
98	15	0.1356	0.7555
99	15	0.1343	0.7562
100	15	0.1331	0.7569

Appendix B

Improved Online Algorithms for Knapsack and GAP in the Random Order Model

Bibliographic information S. Albers, A. Khan, and L. Ladewig. Improved online algorithms for knapsack and GAP in the random order model. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM)*, volume 145 of *LIPIcs*, pages 22:1–22:23. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

Summary We study the online knapsack problem in the random-order model, introduced as the *knapsack secretary problem* by Babaioff et al. [APPROX'07]. Our main contribution is a (1/6.65)-competitive randomized algorithm, outperforming the previously best algorithm by Kesselheim et al. [SIAM J. Comp. 47(5)].

Our algorithm is based on a new algorithmic approach that performs two algorithms, tailored for specific item classes, sequentially. We further investigate the *2-Knapsack problem*, which arises as the special case where all items consume more than one third of the resource's capacity. For 2-Knapsack, we propose a simple algorithm based on our recent progress on the k-secretary problem [ISAAC'19]. Our overall algorithm is composed of the latter algorithm for 2-Knapsack and an algorithm by Kesselheim et al. [SIAM J. Comp. 47(5)] for the generalized assignment problem.

Finally, we discuss the generalized assignment problem and show that our sequential approach again improves the state-of-the-art competitive ratio.

Individual contributions

- Proposal and analysis of the sequential approach from Section 3
- Development of the analysis for the 2-Knapsack problem from Section 4 and the overall analysis from Section 5
- Composition of the manuscript including all technical and non-technical parts (refined based on discussions with co-authors)

Improved Online Algorithms for Knapsack and GAP in the Random Order Model

Susanne Albers
Technical University of Munich, Germany
albers@in.tum.de

Arindam Khan
Indian Institute of Science, Bangalore, India[1]
arindamkhan@iisc.ac.in

Leon Ladewig
Technical University of Munich, Germany
ladewig@in.tum.de

Abstract

The *knapsack problem* is one of the classical problems in combinatorial optimization: Given a set of items, each specified by its size and profit, the goal is to find a maximum profit packing into a knapsack of bounded capacity. In the online setting, items are revealed one by one and the decision, if the current item is packed or discarded forever, must be done immediately and irrevocably upon arrival. We study the online variant in the random order model where the input sequence is a uniform random permutation of the item set.

We develop a randomized (1/6.65)-competitive algorithm for this problem, outperforming the current best algorithm of competitive ratio 1/8.06 [Kesselheim et al. SIAM J. Comp. 47(5)]. Our algorithm is based on two new insights: We introduce a novel algorithmic approach that employs two given algorithms, optimized for restricted item classes, sequentially on the input sequence. In addition, we study and exploit the relationship of the knapsack problem to the 2-secretary problem.

The *generalized assignment problem* (GAP) includes, besides the knapsack problem, several important problems related to scheduling and matching. We show that in the same online setting, applying the proposed sequential approach yields a (1/6.99)-competitive randomized algorithm for GAP. Again, our proposed algorithm outperforms the current best result of competitive ratio 1/8.06 [Kesselheim et al. SIAM J. Comp. 47(5)].

2012 ACM Subject Classification Theory of computation → Online algorithms

Keywords and phrases Online algorithms, knapsack problem, random order model

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.22

Category APPROX

Funding Work supported by the European Research Council, Grant Agreement No. 691672.

1 Introduction

Many real-world problems can be considered resource allocation problems. For example, consider the loading of cargo planes with (potential) goods of different weights. Each item raises a certain profit for the airline if it is transported; however, not all goods can be loaded due to airplane weight restrictions. Clearly, the dispatcher seeks for a maximum profit packing fulfilling the capacity constraint. This example from [24] illustrates the *knapsack problem*: Given a set of n items, specified by a size and a profit value, and a resource (called knapsack) of fixed capacity, the goal is to find a subset of items (called packing) with maximum total

[1] A part of this work was done when the author was at Technical University of Munich.

profit and whose total size does not exceed the capacity. Besides being a fundamental and extensively studied problem in combinatorial optimization, knapsack problems arise in many and various practical settings. We refer the readers to textbooks [24, 35] and to the surveys of previous work in [14, 19] for further references.

In the *generalized assignment problem* (GAP) [35], resources of different capacities are given, and the size and the profit of an item depend on the resource to which it is assigned. The GAP includes many prominent problems, such as the (multiple) knapsack problem [13], weighted bipartite matching [28], AdWords [36], and the display ads problem [17]. Further applications of GAP are outlined in the survey articles [11, 41].

We study online variants of the knapsack and GAP problems. Here, n items are presented sequentially, and the decision for each item must be made immediately upon arrival. In fact, many real-world optimization problems occur as online problems, as often decisions must be made under uncertain conditions. For example, consider the introducing logistics example, if the airline needs to answer customer requests immediately without knowing future requests. The online knapsack problem has been studied in particular in the context of online auctions [9, 45].

Typically, the performance measure for online algorithms is the *competitive ratio*, which is defined as the ratio between the values of the algorithmic solution and an optimal offline solution for a worst-case input. It can be shown that, even for the knapsack problem, the general online setting admits no algorithms with bounded competitive ratio [34, 45]. However, most hardness results are based on a worst-case input presented in adversarial order. In the *random order model*, the performance of an algorithm is evaluated for a worst-case input, but the adversary has no control over the input order; the input sequence is drawn uniformly at random among all permutations. This model is known from the secretary problem [15, 31] and its generalizations [7, 12, 18]; it has been successfully applied to other online problems, for example, scheduling and packing [1, 16, 20, 25, 27, 39], graph problems [8, 26, 33], facility location [37], budgeted allocation [38], and submodular welfare maximization [30].

1.1 Related Work

Online knapsack problem. The problem was first studied by Marchetti-Spaccamela and Vercellis [34], who showed that no deterministic online algorithm for this problem can obtain a constant competitive ratio. Moreover, Chakrabarty et al. [45] demonstrated that this fact cannot be overcome by randomization.

Given such hardness results, several relaxations have been introduced and investigated. Most relevant to our work are results in the random order model. Introduced as the *secretary knapsack problem* [6], Babaioff et al. developed a randomized algorithm of competitive ratio $1/(10e) < 1/27$. Kesselheim et al. [27] achieved a significant improvement by developing a $(1/8.06)$-competitive randomized algorithm for the generalized assignment problem. Finally, Vaze [43] showed that there exists a deterministic algorithm of competitive ratio $1/(2e) < 1/5.44$, assuming that the maximum profit of a single item is small compared to the profit of the optimal solution.

Apart from the random order model, different further relaxations have been considered. Marchetti-Spaccamela and Vercellis [34] studied a stochastic model wherein item sizes and profits are drawn from a fixed distribution. Lueker [32] obtained improved bounds in this model. Chakrabarty et al. [45] studied the problem when the density (profit-size ratio) of each item is in a fixed range $[L, U]$. Under the further assumption that item sizes are small compared to the knapsack capacity, Chakrabarty et al. proposed an algorithm of competitive ratio $\ln(U/L) + 1$ and provided a lower bound of $\ln(U/L)$. Another branch of

research considers removable models, where the algorithm can remove previously packed items. Removing such items can incur no cost [22,23] or a cancellation cost (*buyback model*, [4,5,21]). Recently, Vaze [44] considered the problem under a (weaker) expected capacity constraint. This variant admits a competitive ratio of $1/4e$.

Online GAP. Since all hardness results for online knapsack also hold for online GAP, research focuses on stochastic variants or modified online settings. Currently, the only result for the random order model is the previously mentioned (1/8.06)-competitive randomized algorithm proposed by Kesselheim et al. [27]. To the best of our knowledge, the earliest paper considering online GAP is due to Feldman et al. [17]. They obtained an algorithm of competitive ratio tending to $1 - 1/e$ in the *free disposal model*. In this model, the total size of items assigned to a resource might exceed its capacity; in addition, no item consumes more than a small fraction of any resource. A stochastic variant of online GAP was studied by Alaei et al. [2]. Here, the size of an item is drawn from an individual distribution that is revealed upon arrival of the item, together with its profit. However, the algorithm learns the actual item size only after the assignment. If no item consumes more than a $(1/k)$-fraction of any resource, the algorithm proposed by Alaei et al. has competitive ratio $1 - 1/\sqrt{k}$.

Online packing LPs. In contrast to GAP, general packing LPs describe problems where requests can consume more than one resource. The study of online packing LPs was initiated by Buchbinder and Naor [10] in the adversarial model. In several papers [1, 16, 27, 39] it has been shown that the random order model admits $(1 - \varepsilon)$-competitive algorithms assuming large capacity ratios, i.e., when the capacity of any resource is large compared to the maximum demand for it. Most recently, Kesselheim et al. [27] showed that there is a $(1 - \varepsilon)$-competitive algorithm if $B = \Omega((\log d)/\varepsilon^2)$, where B is the capacity ratio and d is the column sparsity (the maximum number of resources occurring in a single column).

1.2 Our Contributions

As outlined above, for online knapsack and GAP in the adversarial input model, nearly all previous works attain constant competitive ratios at the cost of either (a) imposing structural constraints on the input or (b) significantly relaxing the original online model. Therefore, we study both problems in the random order model, which is less pessimistic than the adversarial model but still considers worst-case instances without further constraints on the item properties. For the knapsack problem, our main result is the following.

▶ **Theorem 1.1.** *There exists a* (1/6.65)*-competitive randomized algorithm for the online knapsack problem in the random order model assuming* $n \to \infty$.

One challenge in the design of knapsack algorithms is that the optimal packing can have, on a high level, at least two different structures. Either there are few large items, constituting the majority of the packing's profit, or there are many small such items. Previous work [6,27] is based on splitting the input according to item sizes and then employing algorithms tailored for these restricted instances. However, the algorithms from [6,27] choose a single item type via an initial random choice, and then pack items of that type exclusively. In contrast, our approach considers different item types in distinct time intervals, rather than discarding items of a specific type in advance. More precisely, we develop algorithms $\mathcal{A}_L$ and $\mathcal{A}_S$ which are combined in a novel *sequential approach*: While large items appearing in early rounds are packed using $\mathcal{A}_L$, algorithm $\mathcal{A}_S$ is applied to pack small items revealed in later rounds. We think that this approach may be helpful for other problems in similar online settings as well.

The proposed algorithm $\mathcal{A}_L$ deals with the knapsack problem where all items consume more than $1/3$ of the capacity (we call this problem 2-KS). The 2-KS problem is closely related to the k-secretary problem [29] for $k = 2$. We also develop a general framework that allows to employ any algorithm for the 2-secretary problem to obtain an algorithm for 2-KS. As a side product, we obtain a simple $(1/3.08)$-competitive deterministic algorithm for 2-KS in the random order model. For items whose size is at most $1/3$ of the resource capacity, we give a simple and efficient algorithm $\mathcal{A}_S$. Here, a challenging constraint is that $\mathcal{A}_L$ and $\mathcal{A}_S$ share the same resource, so we need to argue carefully that the decisions of $\mathcal{A}_S$ are feasible, given the packing of $\mathcal{A}_L$ from previous rounds.

Finally, we show that the proposed sequential approach also improves the current best result for GAP [27] from competitive ratio $1/8.06$ to $1/6.99$.

▶ **Theorem 1.2.** *There exists a* $(1/6.99)$*-competitive randomized algorithm for the online generalized assignment problem in the random order model assuming* $n \to \infty$*.*

For this problem we use the algorithmic building blocks $\mathcal{A}_L$, $\mathcal{A}_S$ developed in [26,27]. However, we need to verify that $\mathcal{A}_L$, an algorithm for edge-weighted bipartite matching [26], satisfies the desired properties for the sequential approach. We point out that the assignments of our algorithm differ structurally from the assignments of the algorithm proposed in [27]. In the assignments of the latter algorithm, all items are either large or small compared to the capacity of the assigned resource. In our approach, both situations can occur, because resources are managed independently.

Roadmap. We focus on the result on the knapsack problem (Theorem 1.1) in the first chapters of this paper. For this purpose, we provide elementary definitions in Section 2. Our main technical contribution is formally introduced in Section 3: Here, we describe an algorithmic framework performing two algorithms $\mathcal{A}_L$, $\mathcal{A}_S$ sequentially. In Sections 4 and 5, we design and analyze the algorithms $\mathcal{A}_L$ and $\mathcal{A}_S$ for the knapsack problem. Finally, in Section 6 we describe how the sequential approach can be applied to GAP. Due to space constraints, some proofs are deferred to Appendix A (knapsack) and to Appendix B (GAP).

2 Preliminaries

Let $[n] := \{1, \dots, n\}$. Further, let $\mathbb{Q}_{\geq 0}$ and $\mathbb{Q}_{>0}$ denote the set of non-negative and positive rational numbers, respectively.

Knapsack problem. We are given a set of items $I = [n]$, each item $i \in I$ has *size* $s_i \in \mathbb{Q}_{>0}$ and a *profit* (*value*) $v_i \in \mathbb{Q}_{\geq 0}$. The goal is to find a maximum profit packing into a knapsack of size $W \in \mathbb{Q}_{>0}$, i.e., a subset $M \subseteq I$ such that $\sum_{i \in M} s_i \leq W$ and $\sum_{i \in M} v_i$ is maximized. W.l.o.g. we can assume $s_i \leq W$ for all $i \in I$. In the online variant of the problem, in each *round* $\ell \in [n]$ a single item i is revealed together with its size and profit. The online algorithm must decide immediately and irrevocably whether to pack i. We call an item *visible in round* ℓ if it arrived in round ℓ or earlier.

Random order performance. We analyze the performance of algorithms in the *random order model*. Given a worst case input $\mathcal{I}$, the order in which $\mathcal{I}$ is presented is drawn uniformly at random from the set of all permutations. For an algorithm $\mathcal{A}$, its *competitive ratio* is defined as $\mathbf{E}[\mathcal{A}(\mathcal{I})] / \mathrm{OPT}(\mathcal{I})$, where $\mathcal{A}(\mathcal{I})$ and $\mathrm{OPT}(\mathcal{I})$ denote the profits of the solutions of $\mathcal{A}$ and an optimal offline algorithm, respectively. Here, the expectation is taken over

Algorithm 1 Sequential approach.

Input : Random permutation π of n items in I, a knapsack of capacity W, parameters $c, d \in (0,1)$ with $c < d$, algorithms $\mathcal{A}_L$, $\mathcal{A}_S$.
Output : A feasible (integral) knapsack packing.
Let ℓ be the current round.
if $\ell \leq cn$ **then**
  Sampling phase – discard all items;
if $cn + 1 \leq \ell \leq dn$ **then**
  Pack $\pi(\ell)$ iff $\mathcal{A}_L$ packs $\pi_L(\ell)$;
if $dn + 1 \leq \ell \leq n$ **then**
  Pack $\pi(\ell)$ iff $\mathcal{A}_S$ packs $\pi_S(\ell)$ and the remaining capacity is sufficiently large.

all permutations and random choices of the algorithm. As above, we slightly overload the notation and also use $\mathcal{A}$ as a random variable for the profit of the solution returned by an algorithm $\mathcal{A}$.

We classify items as large or small, depending on their size compared to W and a parameter $\delta \in (0,1)$ to be determined later.

▶ **Definition 2.1.** *We say an item i is δ-large if $s_i > \delta W$ and δ-small if $s_i \leq \delta W$. Whenever δ is clear from the context, we say an item is large or small for short. Based on the given item set I, we define two modified item sets I_L and I_S, which are obtained as follows:*

- *I_L: Replace each small item by a large item of profit 0*
- *I_S: Replace each large item by a small item of profit 0.*

Therefore, I_L only contains large items and I_S only contains small items. We can assume that no algorithm packs a zero-profit item, thus any algorithmic packing of I_L or I_S can be turned into a packing of I having the same profit. Let OPT, OPT_L, and OPT_S be the total profits of optimal packings for I, I_L, and I_S, respectively. A useful upper bound for OPT is

$$\mathrm{OPT} \leq \mathrm{OPT}_L + \mathrm{OPT}_S\,. \tag{1}$$

3 Sequential Approach

A common approach in the design of algorithms for secretary problems is to set two phases: a *sampling phase*, where all items are rejected, followed by a *decision phase*, where some items are accepted according to a decision rule. Typically, this rule is based on the information gathered in the sampling phase. We take this concept a step further: The key idea of our sequential approach is to use a part of the sampling phase of one algorithm as decision phase of another algorithm, which itself can have a sampling phase. This way, two algorithms are performed in a sequential way, which makes better use of the entire instance. We combine this idea with using different strategies for small and large items.

Formally, let $\mathcal{A}_L$ and $\mathcal{A}_S$ be two online knapsack algorithms and I_L and I_S be the item sets constructed according to Definition 2.1. Further, let $0 < c < d < 1$ be two parameters to be specified later. Our proposed algorithm samples the first cn rounds; during this time no item is packed. From round $cn + 1$ to dn, the algorithm considers large items exclusively. In this interval we follow the decisions of $\mathcal{A}_L$. After round dn, the algorithm processes only small items and follows the decisions of $\mathcal{A}_S$. However, it might be the case that an item accepted by $\mathcal{A}_S$ cannot be packed because the knapsack capacity is exhausted due to the packing of $\mathcal{A}_L$ in earlier rounds. Note that all rounds $1, \ldots, dn$ can be considered as the

Algorithm 2 Algorithm $\mathcal{A}_L$ for large items.

Input : Random permutation of n $(1/3)$-large items, a knapsack of capacity W, parameters $c, d \in (0,1)$ with $c < d$.
Output : A feasible (integral) packing of the knapsack.
Let ℓ be the current round.
if $\ell \leq cn$ **then**
  Sampling phase – discard all items.
Let v^* be the maximum profit seen up to round cn.
if $cn + 1 \leq \ell \leq dn$ **then**
  Pack the first two items of profit higher than v^*, if feasible.
if $\ell > dn$ **then**
  Discard all items.

sampling phase for $\mathcal{A}_S$. A formal description is given in Algorithm 1. Here, for a given input sequence π of I, let π_L and π_S denote the corresponding sequences from I_L and I_S, respectively. Note that π is revealed sequentially and π_L, π_S can be constructed online. For any input sequence π, let $\pi(\ell)$ denote the item at position $\ell \in [n]$.

In the final algorithm we set the threshold for small items to $\delta = 1/3$ and use Algorithm 1 with parameters $c = 0.42291$ and $d = 0.64570$. Under the assumption $n \to \infty$ we can assume $cn, dn \in \mathbb{N}$. We next give a high-level description of the proof of Theorem 1.1.

Proof of Theorem 1.1. Let $\mathcal{A}$ be Algorithm 1 and $\mathcal{A}_L$, $\mathcal{A}_S$ be the algorithms developed in Sections 4 and 5. In the next sections we prove the following results (see Lemmas 4.7 and 5.5): The expected profit from $\mathcal{A}_L$ in rounds $cn+1, \dots, dn$ is at least $\frac{1}{6.65}\,\mathrm{OPT}_L$, and the expected profit from $\mathcal{A}_S$ in rounds $dn+1, \dots, n$ is at least $\frac{1}{6.65}\,\mathrm{OPT}_S$. Together with inequality (1), we obtain

$$\mathbf{E}[\mathcal{A}] \geq \mathbf{E}[\mathcal{A}_L] + \mathbf{E}[\mathcal{A}_S] \geq \frac{1}{6.65}\,\mathrm{OPT}_L + \frac{1}{6.65}\,\mathrm{OPT}_S \geq \frac{1}{6.65}\,\mathrm{OPT}\,. \qquad \blacktriangleleft$$

The order in which $\mathcal{A}_L$ and $\mathcal{A}_S$ are arranged in Algorithm 1 follows from two observations. Algorithm $\mathcal{A}_S$ is powerful if it samples roughly $(2/3)n$ rounds; a part of this long sampling phase can be used as the decision phase of $\mathcal{A}_L$, for which a shorter sampling phase is sufficient. Moreover, the first algorithm should either pack high-profit items, or should leave the knapsack empty for the following algorithm with high probability. The algorithm $\mathcal{A}_L$ we propose in Section 4 has this property (see Lemma 4.8). In contrast, if $\mathcal{A}_S$ would precede $\mathcal{A}_L$, the knapsack would be empty at the beginning of $\mathcal{A}_L$ with very small probability, in which case we would not benefit from $\mathcal{A}_L$.

Finally, note that better algorithms and parameterizations for the respective sub-problems exist (see Lemma 4.6 and [27]). However, for the overall performance we need algorithms $\mathcal{A}_L$ and $\mathcal{A}_S$ that perform well evaluated in the sequential framework.

4 Large Items

The approach presented in this section is based on the connection between the online knapsack problem under random arrival order and the k-secretary problem [29]. In the latter problem, the algorithm can accept up to k items and the goal is to maximize the sum of their profits. The k-secretary problem generalizes the classical secretary problem [15, 31] and is itself a special case of the online knapsack problem under random arrival order (if all knapsack items have size W/k).

Table 1 Definition of packing types A-M. We use set notation $\{i, j\}$ if i and j can be packed in any order, and tuple notation (i, j) if the packing order must be as given.

type	content	constraint on j	probability p_X
A	$\{1, 2\}$	-	$p_{12} + p_{21}$
B	$\{1, 3\}$	-	$p_{13} + p_{31}$
C	$\{2, 3\}$	-	$p_{23} + p_{32}$
D	$(1, j)$	-	p_1
E	$(2, j)$	-	p_2
F	$(3, j)$	-	p_3
G	$(4, j)$	-	p_4
H	$(1, j)$	$j \neq 2$	$p_1 - p_{12}$
I	$(1, j)$	$j \neq 3$	$p_1 - p_{13}$
J	$(2, j)$	$j \neq 1$	$p_2 - p_{21}$
K	$(2, j)$	$j \neq 3$	$p_2 - p_{23}$
L	$(3, j)$	$j \neq 1$	$p_3 - p_{31}$
M	$(3, j)$	$j \neq 2$	$p_3 - p_{32}$

In our setting, each large item consumes more than $\delta = 1/3$ of the knapsack capacity. We call this problem 2-KS, since at most two items can be packed completely. Therefore, any 2-secretary algorithm can be employed to identify high-profit items and pack them if feasible. Although this idea applies to any δ and corresponding k, the approach seems stronger for small k: Intuitively, the characteristics of k-KS and k-secretary deviate with growing k, while 1-KS is exactly 1-secretary. Furthermore, the k-secretary problem is for $k = 2$ rather well studied [3, 12], while the exact optimal competitive ratios for $k \geq 3$ are still unknown.

In the following, let $\mathcal{A}_L$ be Algorithm 2. This is an adaptation of the algorithm SINGLE-REF developed for the k-secretary problem in [3]. As discussed above, 2-secretary and 2-KS are similar, but different problems. Therefore, in our setting it is not possible to apply the existing analysis from [3] or from any other k-secretary algorithm directly.

Assumption. For this section we assume that all profits are distinct. This is without loss of generality, as ties can be broken by adjusting the profits slightly, using the items' identifiers. Further, we assume $v_1 > v_2 > \ldots > v_n$ and say that i is the *rank* of item i.

4.1 Packing Types

As outlined above, in contrast to the 2-secretary problem, not all combinations of two knapsack items can be packed completely. Therefore, we analyze the probability that $\mathcal{A}_L$ selects a feasible set of items whose profit can be bounded from below. We restrict our analysis to packings where an item $i \in \{1, 2, 3, 4\}$ is packed as the first item and group such packings into several packing types A-M defined in the following. Although covering more packings might lead to further insights into the problem and to a stronger result, we expect the improvement to be marginal.

Let p_X be the probability that $\mathcal{A}_L$ returns a packing of type $X \in \{\mathrm{A}, \ldots, \mathrm{M}\}$. In addition, let p_i for $i \in [n]$ be the probability that $\mathcal{A}_L$ packs i as the first item. Finally, let p_{ij} for $i, j \in [n]$ be the probability that $\mathcal{A}_L$ packs i as the first item and j as the second item.

In a packing of type A, the items 1 and 2 are packed in any order. Therefore, $p_{\mathrm{A}} = p_{12} + p_{21}$. The types B and C are defined analogously using the items $\{1, 3\}$ and $\{2, 3\}$, respectively. In a packing of type D, the item 1 is accepted as the first item, together with no or any second

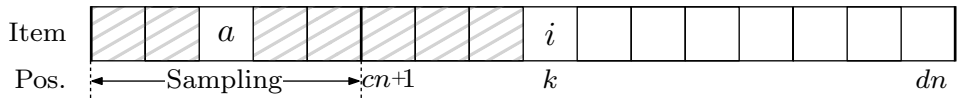


Figure 1 Input sequence considered in Lemma 4.2. The gray dashed slots represent items of rank greater than a.

item j. This happens with probability $p_{\mathrm{D}} = p_1$. Accordingly, we define types E,F, and G using the items 2,3, and 4, respectively. Finally, for each item $i \in \{1,2,3\}$, we introduce two further packing types. For $i = 1$, types H and I characterize packings where the first accepted item is 1, the second accepted item j is not 2 (type H) and not 3 (type I), respectively. Therefore, we get $p_{\mathrm{H}} = p_1 - p_{12}$ and $p_{\mathrm{I}} = p_1 - p_{13}$. Packing types J-K and L-M describe analogous packings for $i = 2$ and $i = 3$, respectively. Table 1 shows all packing types A-M and their probabilities expressed by p_i and p_{ij}.

The packing types defined above allow to describe all packings where a specific item $i \in \{1,2,3,4\}$ is packed as the first item, without covering the same packing multiple times. For example, packing types A and D (with $j = 2$) both include the packing $(1,2)$; however, we can consider the disjoint packing types A and H.

4.2 Acceptance Probabilities of Algorithm 2

In the following we compute the probabilities p_i and p_{ij} from Table 1 as functions of c and d. Throughout the following proofs, we denote the position of an item i in a given permutation with $\mathrm{pos}(i) \in [n]$. Further, let a be the maximum profit item from sampling.

We think of the random permutation as being sequentially constructed. The fact given below follows from the hypergeometric distribution and becomes helpful in the proofs of Lemmas 4.2 and 4.3.

▶ **Fact 4.1.** *Suppose there are N balls in an urn from which M are blue and $N - M$ red. The probability of drawing K blue balls without replacement in a sequence of length K is $h(N,M,K) := \binom{M}{K}/\binom{N}{K}$.*

In the first lemma, we provide the probabilities p_i for $i \in [4]$ assuming $n \to \infty$.

▶ **Lemma 4.2.** *Assuming $n \to \infty$, it holds that*

$$p_i = \begin{cases} c \ln \frac{d}{c} & i = 1 \\ c\left(\ln \frac{d}{c} - d + c\right) & i = 2 \\ c\left(\ln \frac{d}{c} - 2(d-c) + \frac{1}{2}(d^2 - c^2)\right) & i = 3 \\ c\left(\ln \frac{d}{c} - 3(d-c) + \frac{3}{2}(d^2 - c^2) - \frac{1}{3}(d^3 - c^3)\right) & i = 4\,. \end{cases}$$

Proof. We construct the random permutation by drawing the positions for items sequentially, starting with the items i and a. For any position $k \geq cn + 1$, the permutation fulfills $\mathrm{pos}(i) = k$ and $\mathrm{pos}(a) \leq cn$ with probability $\frac{1}{n}\frac{cn}{n-1} = \frac{c}{n-1}$. Next, we draw the remaining $k - 2$ items for the slots up to position k. Since i is packed as the first item, all previous items (except for a) must have rank greater than a (see Figure 1). As these items are drawn from the remaining $n - 2$ items (of which $n - a$ have rank greater than a), the probability for this step is $h(n-2, n-a, k-2)$ according to Fact 4.1. Using the law of total probability for $k \in \{cn+1, \ldots, dn\}$ and $a \in \{i+1, \ldots, n\}$ we obtain

$$p_i = \frac{c}{n-1} \sum_{k=cn+1}^{dn} \sum_{a=i+1}^{n} h(n-2, n-a, k-2) = \frac{c}{n-1} \sum_{k=cn+1}^{dn} \frac{1}{\binom{n-2}{k-2}} \sum_{a=i+1}^{n} \binom{n-a}{k-2}.$$

We can simplify this term further by observing

$$\sum_{a=i+1}^{n} \binom{n-a}{k-2} = \sum_{a=0}^{n-i-1} \binom{a}{k-2} = \binom{n-i}{k-1}.$$

Therefore, $p_i = \frac{c}{n-1} \sum_{k=cn+1}^{dn} \binom{n-i}{k-1} / \binom{n-2}{k-2}$.

Asymptotics. It holds that

$$\lim_{n\to\infty} \frac{\binom{n-i}{k-1}}{\binom{n-2}{k-2}} = \lim_{n\to\infty} \frac{(n-i)!}{(n-2)!} \frac{(n-k)!}{(n-i-k+1)!} \frac{1}{k-1} = \frac{(n-k)^{i-1}}{n^{i-2}} \frac{1}{k}.$$

Hence, $\lim_{n\to\infty} p_i = (c/n^{i-1}) \sum_{k=cn+1}^{dn} f(k)$ where $f(k) := (n-k)^{i-1}/k$. Since f is monotonically decreasing in k, we have $\int_{cn+1}^{dn+1} f(k)\,\mathrm{d}k \leq \sum_{k=cn+1}^{dn} f(k) \leq \int_{cn}^{dn} f(k)\,\mathrm{d}k$. Let F be a function such that $\int_a^b f(k)\,\mathrm{d}k = F(b) - F(a)$ for $0 < a < b$. As it holds that $\lim_{n\to\infty} F(dn+1) - F(dn) = \lim_{n\to\infty} F(cn+1) - F(cn) = 0$, the above bounds are asymptotically tight, i.e., $\lim_{n\to\infty} \sum_{k=cn+1}^{dn} f(k) = F(dn) - F(cn)$. Below we give functions F for $i \in [4]$.

i	$f(k)$	$F(k)$	$F(dn) - F(cn)$
1	$\frac{1}{k}$	$\ln k$	$\ln \frac{d}{c}$
2	$\frac{n-k}{k}$	$n \ln k - k$	$n \ln \frac{d}{c} - dn + cn$
3	$\frac{(n-k)^2}{k}$	$n^2 \ln k - 2nk + \frac{k^2}{2}$	$n^2 \ln \frac{d}{c} - 2n(dn - cn) + \frac{d^2n^2 - c^2n^2}{2}$
4	$\frac{(n-k)^3}{k}$	$n^3 \ln k - 3n^2k + \frac{3}{2}nk^2 - \frac{k^3}{3}$	$n^3 \ln \frac{d}{c} - 3n^3(d-c) + \frac{3}{2}n^3(d^2 - c^2) - \frac{1}{3}n^3(d^3 - c^3)$

The claims follow by multiplying the respective terms with c/n^{i-1}. ◀

Next, we analyze the probabilities p_{ij} for $i \neq j$ and $i, j \in [3]$. The next lemma deals with the cases where $j = i + 1$.

▶ **Lemma 4.3.** *For $n \to \infty$ it holds that*

$$p_{12} = c\left(d - c \ln \frac{d}{c} - c\right),$$
$$p_{23} = c\left(d - c \ln \frac{d}{c} - c - \frac{d^2}{2} + cd - \frac{c^2}{2}\right).$$

The proof of Lemma 4.3 is technically similar to the proof of Lemma 4.2 and thus deferred to Appendix A. It remains to analyze the probabilities p_{13}, p_{31}, p_{21}, and p_{32}. Interestingly, they all reduce to the two probabilities considered in Lemma 4.3. The following two lemmas should be intuitively clear from the description of Algorithm 2. For completeness, we give formal proofs in Appendix A.

▶ **Lemma 4.4.** *For any two items i and j it holds that $p_{ij} = p_{ji}$.*

▶ **Lemma 4.5.** *For any three items $i < k < j$ it holds that $p_{ij} = p_{kj}$.*

Therefore, we have $p_{13} = p_{23}$ by Lemma 4.5 and $p_{31} = p_{13}$, $p_{21} = p_{12}$, and $p_{32} = p_{23}$ by Lemma 4.4.

4.3 Analysis

Let T be the set of items in the optimal packing of I_L. This set may contain a single item, may be a two-item subset of $\{1,2,3\}$, or may be a two-item subset containing an item $j \geq 4$. In the following we analyze the performance of Algorithm 2 for each case.

Single-item case. Let case 1 be the case where $T = \{1\}$. In case 1, $\mathbf{E}[\mathcal{A}_L] \geq p_{\mathrm{D}} \operatorname{OPT}_L$.

Two-item cases. In cases 2–4, we consider packings of the form $T = \{i,j\}$ with $1 \leq i < j \leq 3$. We define cases 2, 3, and 4 as $T = \{1,2\}$, $T = \{1,3\}$, and $T = \{2,3\}$, respectively. We want to consider all algorithmic packings whose profit can be bounded in terms of $\operatorname{OPT}_L = v_i + v_j$. For this purpose, for each case 2-4 we build three groups of feasible packing types, according to whether the profit of a packing is OPT_L, at least v_i, or in the interval $(v_i, v_j]$. We ensure that no packing is counted multiple times by (a) choosing appropriate packing types and (b) grouping these packing types in a disjoint way, according to their profit. Let α_w be the probability that the algorithm returns the optimal packing in case $w \in \{2,3,4\}$. It holds that $\alpha_2 = p_{\mathrm{A}}$, $\alpha_3 = p_{\mathrm{B}}$, and $\alpha_4 = p_{\mathrm{C}}$. In addition, let β_w be the probability that an item $k \leq i$ is packed as the first item in case $w \in \{2,3,4\}$. We have $\beta_2 = p_{\mathrm{H}}$, $\beta_3 = p_{\mathrm{I}}$, and $\beta_4 = p_{\mathrm{D}} + p_{\mathrm{K}}$. Finally, let γ_w be the probability that an item k with $i < k \leq j$ is packed as the first item in case $w \in \{2,3,4\}$. It holds that $\gamma_2 = p_{\mathrm{J}}$, $\gamma_3 = p_{\mathrm{E}} + p_{\mathrm{L}}$, and $\gamma_4 = p_{\mathrm{M}}$.

Finally, we define case 5 as $T = \{i,j\}$ with $i \geq 1$, $j \geq 4$, and $i < j$. In this case, note that packings of type D contain an item of value at least v_i, and packings of type E, F, and G contain an item of value at least v_j. Hence, we can slightly abuse the notation and set $\alpha_5 = 0$, $\beta_5 = p_{\mathrm{D}}$, and $\gamma_5 = p_{\mathrm{E}} + p_{\mathrm{F}} + p_{\mathrm{G}}$, such that it holds that

$$\mathbf{E}[\mathcal{A}_L] \geq \alpha_w(v_i + v_j) + \beta_w v_i + \gamma_w v_j \qquad \text{in case } w \in \{2,3,4,5\}.$$

To bound this term against $\operatorname{OPT}_L = v_i + v_j$, consider the following two cases: If $\beta_w \geq \gamma_w$, we obtain from Chebyshev's sum inequality $\beta_w v_i + \gamma_w v_j \geq \frac{1}{2}(\beta_w + \gamma_w)(v_i + v_j)$. If $\beta_w < \gamma_w$, we trivially have $\beta_w v_i + \gamma_w v_j > \beta_w(v_i + v_j)$. Thus, we obtain

$$\mathbf{E}[\mathcal{A}_L] \geq \left(\alpha_w + \min\left\{\frac{\beta_w + \gamma_w}{2}, \beta_w\right\}\right) \operatorname{OPT}_L \qquad \text{in case } w \in \{2,3,4,5\}. \tag{2}$$

The competitive ratio of $\mathcal{A}_L$ is the minimum over all cases 1-5. We obtain the following two lemmas. If the algorithm is allowed to use the entire input sequence ($d = 1$), $\mathcal{A}_L$ has a competitive ratio of $1/3.08$.

▶ **Lemma 4.6.** *With $c = 0.23053$ and $d = 1$, algorithm $\mathcal{A}_L$ satisfies $\mathbf{E}[\mathcal{A}_L] \geq \frac{1}{3.08} \operatorname{OPT}_L$.*

Note that 2-KS includes the secretary problem (case 1); thus, no algorithm for 2-KS can have a better competitive ratio than $1/e < 1/2.71$. In the final algorithm we set $d < 1$ to benefit from $\mathcal{A}_S$. The next lemma has already been used to prove Theorem 1.1 in Section 3.

▶ **Lemma 4.7.** *With $c = 0.42291$ and $d = 0.64570$, algorithm $\mathcal{A}_L$ satisfies $\mathbf{E}[\mathcal{A}_L] \geq \frac{1}{6.65} \operatorname{OPT}_L$.*

Proof of Lemmas 4.6 and 4.7. For the overall competitive ratio, we build the minimum over all cases. According to inequality (2), the competitive ratios for the two-item cases depend on $\beta_w \geq \gamma_w$ or $\beta_w < \gamma_w$. However, for the parameter pairs $(c,d) = (0.23053, 1)$ from

Table 2 Competitive ratios of Algorithm 2 for the parameters from Lemmas 4.6 and 4.7 in different cases. Bold values indicate the minimum over all cases and thus the competitive ratio.

				two-item cases			
	c	d	case 1	case 2	case 3	case 4	case 5
Lemma 4.6	0.23053	1	0.33827	0.34898	0.32705	0.32705	**0.32471**
Lemma 4.7	0.42291	0.64570	0.17897	**0.15039**	0.16033	0.16033	0.16231

Lemma 4.6 and $(c, d) = (0.42291, 0.64570)$ from Lemma 4.7 we have $\beta_w \geq \gamma_w$ for any case $w \in \{2, 3, 4, 5\}$. This follows from a technical lemma provided in Appendix A (Lemma A.1). Hence, inequality (2) simplifies to $\mathbf{E}[\mathcal{A}_L] \geq \left(\alpha_w + \frac{\beta_w + \gamma_w}{2}\right) \mathrm{OPT}_L$ in case $w \in \{2, 3, 4, 5\}$. Using the definitions of p_X from Table 1 and the symmetry property of Lemma 4.4 we get

$$\mathbf{E}[\mathcal{A}_L] / \mathrm{OPT}_L \geq \begin{cases} p_1 & \text{case 1} \\ p_{12} + (p_1 + p_2)/2 & \text{case 2} \\ p_{13} + (p_1 + p_2 + p_3)/2 & \text{case 3} \\ p_{23} + (p_1 + p_2 + p_3)/2 & \text{case 4} \\ (p_1 + p_2 + p_3 + p_4)/2 & \text{case 5} \end{cases} . \tag{3}$$

Note that the algorithm attains the same competitive ratio in case 3 and 4, since $p_{13} = p_{23}$. Table 2 shows the competitive ratios for all five cases obtained from Equation (3). For the overall competitive ratio, we have

$$\mathbf{E}[\mathcal{A}_L] \geq \min\left\{p_1, p_{12} + \frac{p_1 + p_2}{2}, p_{23} + \frac{p_1 + p_2 + p_3}{2}, \frac{p_1 + p_2 + p_3 + p_4}{2}\right\} \mathrm{OPT}_L .$$

Hence, the competitive ratios are $0.32471 \geq 1/3.08$ and $0.15039 \geq 1/6.65$ for Lemma 4.6 and Lemma 4.7, respectively. ◀

Recall that in Algorithm 1, we can only benefit from $\mathcal{A}_S$ if $\mathcal{A}_L$ has not filled the knapsack completely. Thus, the following property is crucial in the final analysis.

▶ **Lemma 4.8.** *With probability of at least c/d, no item is packed by $\mathcal{A}_L$.*

Proof. Fix any set of dn items arriving in rounds $1, \ldots, dn$. The most profitable item v^* from this set arrives in the sampling phase with probability c/d. If this event occurs, no item in rounds $cn + 1, \ldots, dn$ beats v^* and $\mathcal{A}_L$ will not select any item. ◀

We finally note that our approach from Section 4.1 provides a general framework to obtain algorithms for 2-KS using secretary algorithms with two choices. Although stronger algorithms than Algorithm 2 exist for the 2-secretary objective [3, 12] and similar objectives [40, 42], it is not clear if they would improve the performance of the overall algorithm. More sophisticated algorithms may use weaker thresholds to accept the first item, which decreases the probability considered in Lemma 4.8. This, in turn, reduces the expected profit gained from $\mathcal{A}_S$, as described above.

5 Small Items

For small items, we use solutions for the fractional problem variant and obtain an integral packing via randomized rounding. This approach has been applied successfully to packing LPs [27]; however, for the knapsack problem it is not required to solve LP relaxations in each

round (as in [27]). Instead, here, we build upon solutions of the classical greedy algorithm, which is well-known to be optimal for the fractional knapsack problem. Particularly, this algorithm is both efficient in running time and easy to analyze.

We next formalize the greedy solution for any set T of items. Let the *density* of an item be the ratio of its profit to its size. Consider any list L containing the items from T ordered by non-increasing density. We define the *rank* $\rho(i)$ of item i as its position in L and $\sigma(l)$ as the item at position l in L. Thus, $\sigma(l) = \rho^{-1}(l)$ denotes the l-th densest item. Let k be such that $\sum_{i=1}^{k-1} s_{\sigma(i)} < W \leq \sum_{i=1}^{k} s_{\sigma(i)}$. The fraction of item i in the greedy solution α is now defined as

$$\alpha_i = \begin{cases} 1 & \text{if } \rho(i) < k \\ \left(W - \sum_{i=1}^{k-1} s_{\sigma(i)}\right) / s_i & \text{if } \rho(i) = k \\ 0 & \text{else}\,, \end{cases}$$

i.e., we pack the $k-1$ densest items integrally and fill the remaining space by the maximum feasible fraction of the k-th densest item. Let $\mathrm{OPT}(T)$ and $\mathrm{OPT}^*(T)$ denote the profits of optimal integral and fractional packings of T, respectively. It is not hard to see that α satisfies $\sum_{i \in T} \alpha_i v_i = \mathrm{OPT}^*(T) \geq \mathrm{OPT}(T)$ and $\sum_{i \in T} \alpha_i s_i = W$.

5.1 Algorithm

The algorithm $\mathcal{A}_S$ for small items, which is formally defined in Algorithm 3, works as follows. After a sampling phase of dn rounds, in each round $\ell \geq dn + 1$ the algorithm computes a greedy solution $x^{(\ell)}$ for $I_S(\ell)$. Here, $I_S(\ell)$ denotes the subset of I_S revealed up to round ℓ. The algorithm packs the current online item i with probability $x_i^{(\ell)}$. However, generally, this can only be done if the remaining capacity of the knapsack is at least $\delta W \geq s_i$.

Note that in case of an integral coefficient $x_i^{(\ell)} \in \{0, 1\}$, the packing step is completely deterministic. Moreover, in any greedy solution $x^{(\ell)}$, there is at most one item i with fractional coefficient $x_i^{(\ell)} \in (0, 1)$. Therefore, in expectation, there is only a small number of rounds where the algorithm actually requests randomness.

▶ **Observation 5.1.** *Let X denote the number of rounds where Algorithm 3 packs an item with probability $x_i \in (0, 1)$. It holds that $\mathbf{E}[X] \leq \ln(1/d) \leq 0.44$.*

Proof. Consider any round ℓ and let $x^{(\ell)}$ be the greedy knapsack solution computed by Algorithm 3. By definition of $x^{(\ell)}$, at most one of the ℓ visible items has a fractional coefficient $x_i^{(\ell)} \in (0, 1)$. The probability that this item i arrives in round ℓ is $1/\ell$ in a random permutation. Let X_ℓ be an indicator variable for the event that Algorithm 3 packs an item at random in round ℓ. By the above argument, we have $\mathbf{Pr}[X_\ell = 1] \leq 1/\ell$. Since Algorithm 3 selects items starting in round $dn+1$, we obtain $\mathbf{E}[X] = \sum_{\ell=dn+1}^{n} \mathbf{E}[X_\ell] \leq \sum_{\ell=dn+1}^{n} \frac{1}{\ell} \leq \ln \frac{1}{d} \leq 0.44\,.$ ◀

Note that Algorithm 2 and the sequential approach (Algorithm 1) are deterministic algorithms. Therefore, our overall algorithm requests randomness in expectation in less than one round.

5.2 Analysis

Let α be the greedy (offline) solution for I_S and set $\Delta = \frac{1}{1-\delta}$. Recall that in round $dn + 1$, the knapsack might already have been filled by $\mathcal{A}_L$ with large items in previous rounds. For now, we assume an empty knapsack after round dn and define this event as ξ. In the final analysis, we will use the fact that $\mathbf{Pr}[\xi]$ can be bounded from below, which is according to Lemma 4.8.

Algorithm 3 Algorithm $\mathcal{A}_S$ for small items.

Input : Random permutation of n $(1/3)$-small items, a knapsack of capacity W, parameter $d \in (0,1)$.
Output : A feasible (integral) packing of the knapsack.
Let ℓ be the current round and i be the online item of round ℓ.
if $\ell \leq dn$ **then**
  Sampling phase – discard all items.
if $dn+1 \leq \ell \leq n$ **then**
  Let $x^{(\ell)}$ be the greedy solution for $I_S(\ell)$.
  if *the remaining capacity is at least* δW **then**
    Pack i with probability $x_i^{(\ell)}$.

▶ **Lemma 5.2.** *Let $i \in I_S$ and $E_i(\ell)$ be the event that the item i is packed by $\mathcal{A}_S$ in round ℓ. For $\ell \geq dn+1$, it holds that* $\mathbf{Pr}\left[E_i(\ell) \mid \xi\right] \geq \frac{1}{n}\alpha_i(1-\Delta\ln\frac{\ell}{dn})$.

Proof. In a random permutation, item i arrives in round ℓ with probability $1/n$. In round $\ell \geq dn+1$, the algorithm decides to pack i with probability $x_i^{(\ell)}$. Note that the rank of item i in $I_S(\ell)$ is less or equal to its rank in I_S. According to the greedy solution's definition, this implies $x_i^{(\ell)} \geq \alpha_i$. Finally, the δ-small item i can be packed successfully if the current resource consumption X is at most $(1-\delta)W$. In the following, we investigate the expectation of X to give a probability bound using Markov's inequality at the end of this proof.

Let X_k be the resource consumption in round $k < \ell$. By assumption, the knapsack is empty after round dn, we have $X = \sum_{k=dn+1}^{\ell-1} X_k$. Let Q be the set of k visible items in round k. The set Q can be seen as uniformly drawn from all k-item subsets and any item $j \in Q$ is the current online item of round k with probability $1/k$. The algorithm packs any item j with probability $x_j^{(k)}$, thus

$$\mathbf{E}\left[X_k\right] = \sum_{j\in Q} \mathbf{Pr}\left[j \text{ occurs in round } k\right] s_j x_j^{(k)} = \frac{1}{k}\sum_{j\in Q} s_j x_j^{(k)} \leq \frac{W}{k}\,,$$

where the last inequality holds because $x^{(k)}$ is a feasible solution for a knapsack of size W. By the linearity of expectation and the previous equation, the expected resource consumption up to round ℓ is $\mathbf{E}\left[X\right] = \sum_{k=dn+1}^{\ell-1}\mathbf{E}\left[X_k\right] \leq \sum_{k=dn+1}^{\ell-1}\frac{W}{k} \leq W\ln\frac{\ell}{dn}$. Using Markov's inequality, we obtain finally

$$\mathbf{Pr}\left[X < (1-\delta)W\right] = 1 - \mathbf{Pr}\left[X \geq (1-\delta)W\right] \geq 1 - \frac{\mathbf{E}\left[X\right]}{(1-\delta)W} \geq 1 - \Delta\ln\frac{\ell}{dn}\,. \qquad \blacktriangleleft$$

Using Lemma 5.2 we easily obtain the total probability that a specific item will be packed.

▶ **Lemma 5.3.** *Let $i \in I_S$ and E_i be the event that the item i is packed by $\mathcal{A}_S$. It holds that* $\mathbf{Pr}\left[E_i \mid \xi\right] \geq \alpha_i\left((1-d)(1+\Delta) - \Delta\ln\frac{1}{d}\right)$.

Proof. Summing the probabilities from Lemma 5.2 over all rounds $\ell \geq dn+1$ gives

$$\begin{aligned}\mathbf{Pr}\left[E_i \mid \xi\right] &= \sum_{\ell=dn+1}^{n}\mathbf{Pr}\left[E_i(\ell)\mid\xi\right] \geq \sum_{\ell=dn+1}^{n}\frac{1}{n}\alpha_i\left(1-\Delta\ln\frac{\ell}{dn}\right)\\ &= \frac{1}{n}\alpha_i\left(n - dn - \Delta\sum_{\ell=dn+1}^{n}\ln\frac{\ell}{dn}\right) = \alpha_i\left(1-d-\frac{\Delta}{n}\sum_{\ell=dn+1}^{n}\ln\frac{\ell}{dn}\right).\end{aligned}$$

Since $\ln \frac{\ell}{dn}$ is monotonically increasing in ℓ, we can bound the last sum by the corresponding integral:

$$\sum_{\ell=dn+1}^{n} \ln \frac{\ell}{dn} \leq \int_{\ell=dn+1}^{n+1} \ln \frac{\ell}{dn} \, \mathrm{d}\ell = (n+1) \ln \frac{n+1}{dn} - (n+1) - (dn+1) \ln \frac{dn+1}{dn} + (dn+1) \, .$$

This implies $\lim_{n\to\infty} \frac{\Delta}{n} \sum_{\ell=dn+1}^{n} \ln \frac{\ell}{dn} \leq \Delta \left(\ln \frac{1}{d} - 1 + d \right)$. Rearranging terms gives the claim. ◀

The following lemma bounds the expected profit of the packing of $\mathcal{A}_S$, assuming ξ.

▶ **Lemma 5.4.** *It holds that* $\mathbf{E}\left[\mathcal{A}_S \mid \xi\right] \geq \left((1-d)(1+\Delta) - \Delta \ln \frac{1}{d}\right) \mathrm{OPT}_S$.

Proof. Let $\beta = (1-d)(1+\Delta) - \Delta \ln \frac{1}{d}$. By Lemma 5.3, the probability that an item i gets packed is $\mathbf{Pr}\left[E_i \mid \xi\right] \geq \alpha_i \beta$. Therefore,

$$\mathbf{E}\left[\mathcal{A}_S \mid \xi\right] = \sum_{i \in I_S} \mathbf{Pr}\left[E_i \mid \xi\right] v_i \geq \sum_{i \in I_S} \alpha_i \beta v_i \geq \beta \, \mathrm{OPT}_S \, . \qquad \blacktriangleleft$$

The conditioning on ξ can be resolved using Lemma 4.8. Thus we obtain the following lemma, which is the second pillar in the proof of Theorem 1.1 and concludes this section.

▶ **Lemma 5.5.** *With* $c = 0.42291$ *and* $d = 0.64570$*, we have* $\mathbf{E}\left[\mathcal{A}_S\right] \geq \frac{1}{6.65} \mathrm{OPT}_S$.

Proof. By Lemma 4.8, the probability for an empty knapsack after round dn is $\mathbf{Pr}\left[\xi\right] \geq \frac{c}{d}$. Thus, from Lemma 5.4 with $\Delta = \frac{1}{1-1/3} = \frac{3}{2}$, we obtain

$$\mathbf{E}\left[\mathcal{A}_S\right] = \mathbf{Pr}\left[\xi\right] \mathbf{E}\left[\mathcal{A}_S \mid \xi\right] = \frac{c}{d} \left(\frac{5}{2}(1-d) - \frac{3}{2} \ln \frac{1}{d} \right) \mathrm{OPT}_S \geq \frac{1}{6.65} \mathrm{OPT}_S \, . \qquad \blacktriangleleft$$

6 Extension to GAP

In this section we show that the sequential approach introduced in Section 3 can be easily adapted to GAP, yielding a (1/6.99)-competitive randomized algorithm. We first define the problem formally.

GAP. We are given a set of items $I = [n]$ and a set of resources $R = [m]$ of capacities $W_r \in \mathbb{Q}_{>0}$ for $r \in R$. If item $i \in I$ is assigned to resource $r \in R$, this raises profit (value) $v_{i,r} \in \mathbb{Q}_{\geq 0}$, but consumes $s_{i,r} \in \mathbb{Q}_{>0}$ of the resource's capacity. The goal is to assign each item to at most one resource such that the total profit is maximized and no resource exceeds its capacity. We call the tuple $(v_{i,r}, s_{i,r})$ an *option* of item i and w.l.o.g. assume that options for all resources exist. This can be ensured by introducing dummy options with $v_{i,r} = 0$. In the online version of the problem, in each round an item is revealed together with its set of options. The online algorithm must decide immediately and irrevocably, if the item is assigned. If so, it has to specify the resource according to one of its options.

Again, we construct restricted instances I_L and I_S according to the following definition, which generalizes Definition 2.1. Let $\delta \in (0,1)$.

▶ **Definition 6.1.** *We call an option* $(v_{i,r}, s_{i,r})$ δ*-large if* $s_{i,r} > \delta W_r$ *and* δ*-small if* $s_{i,r} \leq \delta W_r$*. Whenever* δ *is clear from the context, we say an option is large or small for short. Based on a given instance* I *for GAP, we define two modified instances* I_L *and* I_S *which are obtained from* I *as follows.*

- I_L*: Replace each small option* $(v_{i,r}, s_{i,r})$ *by the large option* $(0, W_r)$*.*
- I_S*: Replace each large option* $(v_{i,r}, s_{i,r})$ *by the small option* $(0, \delta)$*.*

Thus, I_L only contains large options and I_S only contains small options. However, by construction no algorithm will assign an item according to a zero-profit option. We define OPT, OPT_L, and OPT_S accordingly. Note that the inequality $\mathrm{OPT} \leq \mathrm{OPT}_L + \mathrm{OPT}_S$ holds also for GAP.

The sequential framework of Algorithm 1 can be adapted in a straightforward manner by replacing terms like *packing* with *assignment to resource* r. Here, we set the threshold parameter to $\delta = 1/2$. In the following subsections, we specify algorithms $\mathcal{A}_L$ and $\mathcal{A}_S$ for $(1/2)$-large and $(1/2)$-small options, respectively.

6.1 Large Options

If each item consumes more than one half of a resource, no two items can be assigned to this resource. Thus, we obtain the following matching problem.

Edge-weighted bipartite matching problem. Given a bipartite graph $G = (L \cup R, E)$ and a weighting function $w\colon E \to \mathbb{Q}_{\geq 0}$, the goal is to find a bipartite matching $M \subseteq E$ such that $w(M) := \sum_{e \in M} w(e)$ is maximal. In the online version, the (offline) nodes from R and the number $n = |L|$ are known in advance, whereas the nodes from L are revealed online together with their incident edges. In the case of GAP, L is the set of items, R is the set of resources, and the weight of an edge $e = \{l, r\}$ is $w(e) = v_{l,r}$, i.e., the profit gained from assigning item l to resource r.

Under random arrival order, Kesselheim et al. [26] developed an optimal $(1/e)$-competitive algorithm for this problem. Adapting this algorithm to the sequential approach with parameters c and d leads to the following algorithm $\mathcal{A}_L$: After sampling the first cn nodes, in each round ℓ the algorithm computes a maximum edge-weighted matching $M^{(\ell)}$ for the graph revealed up to this round. Let $l \in L$ be the online vertex of round ℓ. If l is matched in $M^{(\ell)}$ to some node $r \in R$, we call $e^{(\ell)} = \{l, r\}$ the *tentative edge* of round ℓ. Now, if r is still unmatched and $\ell \leq dn$, the tentative edge is added to the matching.

A formal description of this algorithm is given in Appendix B.1. The proof of the approximation guarantee relies mainly on the following two lemmas; for completeness, we give the proofs from [26] in Appendix B.1. The first lemma shows that the expected weight of any tentative edge can be bounded from below.

▶ **Lemma 6.2** ([26]). *In any round ℓ, the tentative edge (if it exists) has expected weight* $\mathbf{E}\left[w(e^{(\ell)})\right] \geq \frac{1}{n}\,\mathrm{OPT}_L$.

However, we only gain the weight of the tentative edge $e^{(\ell)} = \{l, r\}$ if it can be added to the matching, i.e., if r has not been matched previously. The next lemma bounds the probability for this event from below.

▶ **Lemma 6.3** ([26]). *Let $\xi(r, \ell)$ be the event that the offline vertex $r \in R$ is unmatched after round ℓ. It holds that* $\mathbf{Pr}\left[\xi(r, \ell)\right] \geq \frac{cn}{\ell}$.

Using Lemmas 6.2 and 6.3, we can bound the competitive ratio of $\mathcal{A}_L$ in the following lemma. Note that we obtain the optimal algorithm from [26] for $c = 1/e$ and $d = 1$.

▶ **Lemma 6.4.** *For $n \to \infty$, it holds that* $\mathbf{E}\left[\mathcal{A}_L\right] \geq c \ln \frac{d}{c}\,\mathrm{OPT}_L$.

Proof. Let A_ℓ be the gain of the matching weight in round ℓ. As the tentative edge $e^{(\ell)} = \{l, r\}$ can only be added if r has not been matched in a previous round, we have $\mathbf{E}\left[A_\ell\right] = \mathbf{E}\left[w(e^{(\ell)})\right] \mathbf{Pr}\left[\xi(r, \ell)\right]$ for the event $\xi(r, \ell)$ from Lemma 6.3. Therefore, from

Lemmas 6.2 and 6.3 we have $\mathbf{E}[A_\ell] \geq \frac{1}{n}\operatorname{OPT}_L \frac{cn}{\ell} = \frac{c}{\ell}\operatorname{OPT}_L$. Summing over all rounds from $cn+1$ to dn yields

$$\mathbf{E}[\mathcal{A}_L] = \sum_{\ell=cn+1}^{dn} \mathbf{E}[A_\ell] \geq \left(c \sum_{\ell=cn+1}^{dn} \frac{1}{\ell}\right) \operatorname{OPT}_L \geq c \ln \frac{dn+1}{cn+1} \operatorname{OPT}_L \ .$$

Here, in the last step we used the fact $\sum_{\ell=cn+1}^{dn} \frac{1}{\ell} \geq \int_{cn+1}^{dn+1} \frac{1}{\ell}\,\mathrm{d}\ell = \ln \frac{dn+1}{cn+1}$. The claim follows by $\lim_{n\to\infty} \ln \frac{dn+1}{cn+1} = \ln \frac{d}{c}$. ◀

6.2 Small Options

For δ-small options we use the LP-based algorithm $\mathcal{A}_S$ from [27, Sec. 3.3]. On a high level, this algorithm works as follows: After a sampling phase of dn rounds, in each round ℓ the algorithm computes an optimal fractional solution for the instance revealed so far and uses the coefficients as probabilities for an integral assignment. In Appendix B.2 we prove the following lemma, where $\Delta = \frac{1}{1-\delta}$.

▶ **Lemma 6.5.** *For $n \to \infty$, it holds that $\mathbf{E}[\mathcal{A}_S] \geq \frac{c}{d}\left((1+\Delta)(1-d) - \Delta \ln \frac{1}{d}\right)\operatorname{OPT}_S$.*

Note that we obtain basically the same competitive ratio as in Lemma 5.4. Since Lemma 6.5 already addresses possible resource consumption due to assignments made by $\mathcal{A}_L$ in earlier rounds, the factor c/d arises (see Lemma 6.3).

6.3 Proof of Theorem 1.2

Finally, we prove our main theorem for GAP.

Proof of Theorem 1.2. We set the threshold between large and small options to $\delta = 1/2$ and consider Algorithm 1 with the algorithms $\mathcal{A}_L$ and $\mathcal{A}_S$ as defined previously. By Lemma 6.4, the expected gain of profit in rounds $cn+1, \ldots, dn$ is $\mathbf{E}[\mathcal{A}_L] \geq c \ln \frac{d}{c} \operatorname{OPT}_L$. Further, we gain $\mathbf{E}[\mathcal{A}_S] \geq \frac{c}{d}\left((1+\Delta)(1-d) - \Delta \ln \frac{1}{d}\right)\operatorname{OPT}_S$ with $\Delta = 2$ in the following rounds, according to Lemma 6.5. For parameters $c = 0.5261$ and $d = 0.6906$, we obtain $c \ln \frac{d}{c} \geq \frac{c}{d}\left(3(1-d) - 2\ln \frac{1}{d}\right)$ and thus, using $\operatorname{OPT}_L + \operatorname{OPT}_S \geq \operatorname{OPT}$,

$$\mathbf{E}[\mathcal{A}_L] + \mathbf{E}[\mathcal{A}_S] \geq \frac{c}{d}\left(3(1-d) - 2\ln \frac{1}{d}\right)(\operatorname{OPT}_L + \operatorname{OPT}_S) \geq \frac{1}{6.99}\operatorname{OPT} \ . \qquad ◀$$

References

1 Shipra Agrawal, Zizhuo Wang, and Yinyu Ye. A Dynamic Near-Optimal Algorithm for Online Linear Programming. *Operations Research*, 62(4):876–890, 2014.

2 Saeed Alaei, MohammadTaghi Hajiaghayi, and Vahid Liaghat. The Online Stochastic Generalized Assignment Problem. In *Proc. 16th International Workshop on Approximation, Randomization, and Combinatorial Optimization and 17th International Workshop on Randomization and Computation (APPROX/RANDOM)*, pages 11–25, 2013.

3 Susanne Albers and Leon Ladewig. New results for the k-secretary problem. Unpublished manuscript, 2018.

4 Moshe Babaioff, Jason Hartline, and Robert Kleinberg. Selling banner ads: Online algorithms with buyback. In *Fourth Workshop on Ad Auctions*, 2008.

5 Moshe Babaioff, Jason D. Hartline, and Robert D. Kleinberg. Selling ad campaigns: online algorithms with cancellations. In *Proc. 10th ACM Conference on Electronic Commerce (EC)*, pages 61–70, 2009.

6 Moshe Babaioff, Nicole Immorlica, David Kempe, and Robert Kleinberg. A Knapsack Secretary Problem with Applications. In *Proc. 10th International Workshop on Approximation, Randomization, and Combinatorial Optimization and 11th International Workshop on Randomization and Computation (APPROX/RANDOM)*, pages 16–28, 2007.

7 Moshe Babaioff, Nicole Immorlica, David Kempe, and Robert Kleinberg. Matroid Secretary Problems. *Journal of the ACM*, 65(6):35:1–35:26, 2018.

8 Bahman Bahmani, Aranyak Mehta, and Rajeev Motwani. A 1.43-Competitive Online Graph Edge Coloring Algorithm in the Random Order Arrival Model. In *Proc. 21st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 31–39, 2010.

9 Christian Borgs, Jennifer T. Chayes, Nicole Immorlica, Kamal Jain, Omid Etesami, and Mohammad Mahdian. Dynamics of bid optimization in online advertisement auctions. In *Proc. 16th International Conference on World Wide Web (WWW)*, pages 531–540, 2007.

10 Niv Buchbinder and Joseph Naor. Online Primal-Dual Algorithms for Covering and Packing. *Math. Oper. Res.*, 34(2):270–286, 2009.

11 Dirk G. Cattrysse and Luk N. Van Wassenhove. A survey of algorithms for the generalized assignment problem. *European Journal of Operational Research*, 60(3):260–272, 1992.

12 T.-H. Hubert Chan, Fei Chen, and Shaofeng H.-C. Jiang. Revealing Optimal Thresholds for Generalized Secretary Problem via Continuous LP: Impacts on Online K-Item Auction and Bipartite K-Matching with Random Arrival Order. In *Proc. 26th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1169–1188, 2015.

13 Chandra Chekuri and Sanjeev Khanna. A Polynomial Time Approximation Scheme for the Multiple Knapsack Problem. *SIAM Journal on Computing (SICOMP)*, 35(3):713–728, 2005.

14 Henrik I. Christensen, Arindam Khan, Sebastian Pokutta, and Prasad Tetali. Approximation and online algorithms for multidimensional bin packing: A survey. *Computer Science Review*, 24:63–79, 2017.

15 Eugene B Dynkin. The optimum choice of the instant for stopping a Markov process. *Soviet Mathematics*, 4:627–629, 1963.

16 Jon Feldman, Monika Henzinger, Nitish Korula, Vahab S. Mirrokni, and Clifford Stein. Online Stochastic Packing Applied to Display Ad Allocation. In *Proc. 18th Annual European Symposium on Algorithms (ESA)*, pages 182–194, 2010.

17 Jon Feldman, Nitish Korula, Vahab S. Mirrokni, S. Muthukrishnan, and Martin Pál. Online Ad Assignment with Free Disposal. In *Proc. 5th International Workshop Internet and Network Economics (WINE)*, pages 374–385, 2009.

18 P.R. Freeman. The secretary problem and its extensions: A review. *International Statistical Review/Revue Internationale de Statistique*, pages 189–206, 1983.

19 Waldo Gálvez, Fabrizio Grandoni, Sandy Heydrich, Salvatore Ingala, Arindam Khan, and Andreas Wiese. Approximating Geometric Knapsack via L-Packings. In *Proc. 58th IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, pages 260–271, 2017.

20 Oliver Göbel, Thomas Kesselheim, and Andreas Tönnis. Online Appointment Scheduling in the Random Order Model. In *Proc. 23rd Annual European Symposium on Algorithms (ESA)*, pages 680–692, 2015.

21 Xin Han, Yasushi Kawase, and Kazuhisa Makino. Online Unweighted Knapsack Problem with Removal Cost. *Algorithmica*, 70(1):76–91, 2014.

22 Xin Han, Yasushi Kawase, and Kazuhisa Makino. Randomized algorithms for online knapsack problems. *Theoretical Computer Science*, 562:395–405, 2015.

23 Kazuo Iwama and Shiro Taketomi. Removable Online Knapsack Problems. In *Proc. 29th International Colloquium on Automata, Languages and Programming (ICALP)*, pages 293–305, 2002.

24 Hans Kellerer, Ulrich Pferschy, and David Pisinger. *Knapsack problems*. Springer, 2004.

25 Claire Kenyon. Best-Fit Bin-Packing with Random Order. In *Proc. 7th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 359–364, 1996.

26 Thomas Kesselheim, Klaus Radke, Andreas Tönnis, and Berthold Vöcking. An Optimal Online Algorithm for Weighted Bipartite Matching and Extensions to Combinatorial Auctions. In *Proc. 21st Annual European Symposium on Algorithms (ESA)*, pages 589–600, 2013.

27 Thomas Kesselheim, Klaus Radke, Andreas Tönnis, and Berthold Vöcking. Primal Beats Dual on Online Packing LPs in the Random-Order Model. *SIAM J. Comput.*, 47(5):1939–1964, 2018.

28 Samir Khuller, Stephen G. Mitchell, and Vijay V. Vazirani. On-Line Algorithms for Weighted Bipartite Matching and Stable Marriages. *Theoretical Computer Science*, 127(2):255–267, 1994.

29 Robert D. Kleinberg. A multiple-choice secretary algorithm with applications to online auctions. In *Proc. 16th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 630–631, 2005.

30 Nitish Korula, Vahab S. Mirrokni, and Morteza Zadimoghaddam. Online Submodular Welfare Maximization: Greedy Beats 1/2 in Random Order. *SIAM J. Comput.*, 47(3):1056–1086, 2018.

31 Denis V Lindley. Dynamic programming and decision theory. *Applied Statistics*, pages 39–51, 1961.

32 George S. Lueker. Average-Case Analysis of Off-Line and On-Line Knapsack Problems. *J. Algorithms*, 29(2):277–305, 1998.

33 Mohammad Mahdian and Qiqi Yan. Online bipartite matching with random arrivals: an approach based on strongly factor-revealing LPs. In *Proc. 43rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 597–606, 2011.

34 Alberto Marchetti-Spaccamela and Carlo Vercellis. Stochastic on-line knapsack problems. *Mathematical Programming*, 68:73–104, 1995.

35 Silvano Martello and Paolo Toth. *Knapsack Problems: Algorithms and Computer Implementations*. John Wiley & Sons, Inc., New York, NY, USA, 1990.

36 Aranyak Mehta, Amin Saberi, Umesh V. Vazirani, and Vijay V. Vazirani. AdWords and generalized online matching. *Journal of the ACM*, 54(5):22, 2007.

37 Adam Meyerson. Online Facility Location. In *Proc. 42nd IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, pages 426–431, 2001.

38 Vahab S. Mirrokni, Shayan Oveis Gharan, and Morteza Zadimoghaddam. Simultaneous approximations for adversarial and stochastic online budgeted allocation. In *Proc. 23rd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1690–1701, 2012.

39 Marco Molinaro and R. Ravi. The Geometry of Online Packing Linear Programs. *Math. Oper. Res.*, 39(1):46–59, 2014.

40 ML Nikolaev. On a generalization of the best choice problem. *Theory of Probability & Its Applications*, 22(1):187–190, 1977.

41 Temel Öncan. A Survey of the Generalized Assignment Problem and Its Applications. *Information Systems and Operational Research INFOR*, 45(3):123–141, 2007.

42 Mitsushi Tamaki. Recognizing both the maximum and the second maximum of a sequence. *Journal of Applied Probability*, 16(4):803–812, 1979.

43 Rahul Vaze. Online knapsack problem and budgeted truthful bipartite matching. In *Proc. IEEE Conference on Computer Communications (INFOCOM) 2017*, pages 1–9, 2017.

44 Rahul Vaze. Online Knapsack Problem Under Expected Capacity Constraint. In *Proc. IEEE Conference on Computer Communications (INFOCOM) 2018*, pages 2159–2167, 2018.

45 Yunhong Zhou, Deeparnab Chakrabarty, and Rajan M. Lukose. Budget Constrained Bidding in Keyword Auctions and Online Knapsack Problems. In *Proc. 4th International Workshop Internet and Network Economics (WINE)*, pages 566–576, 2008.

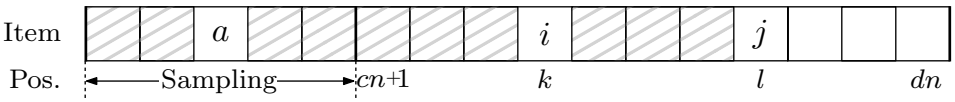


Figure 2 Input sequence considered in Lemma 4.3. The gray dashed slots represent items of rank greater than a.

A Missing Proofs for the Knapsack Result

Proof of Lemma 4.3. Let $i \in [n-1]$ and $j = i+1$. The proof follows the same structure as the proof of Lemma 4.2. Again, we construct the permutation by drawing the positions for items i, j, a first and afterwards all remaining items with position up to $\mathrm{pos}(j)$. Fix positions $k = \mathrm{pos}(i)$ and $l = \mathrm{pos}(j)$. Again, $\mathrm{pos}(a) \le cn$ must hold by definition of a. The probability that a random permutation satisfies these three position constraints is $\beta := \frac{1}{n}\frac{1}{n-1}\frac{cn}{n-2}$. All remaining items up to position l must have rank greater than a (see Figure 2). Thus we need to draw $l-3$ items from a set of $n-3$ remaining items, from which $n-a$ have rank greater than a. This happens with probability $h(n-3, n-a, l-3)$. Using the law of total probability for $cn+1 \le k < l \le dn$ and $a \in \{j+1, \dots, n\}$, we obtain

$$\begin{aligned} p_{ij} &= \beta \sum_{k=cn+1}^{dn-1} \sum_{l=k+1}^{dn} \sum_{a=j+1}^{n} h(n-3, n-a, l-3) \\ &= \beta \sum_{k=cn+1}^{dn-1} \sum_{l=k+1}^{dn} \frac{1}{\binom{n-3}{l-3}} \sum_{a=j+1}^{n} \binom{n-a}{l-3} = \beta \sum_{k=cn+1}^{dn-1} \sum_{l=k+1}^{dn} \frac{\binom{n-j}{l-2}}{\binom{n-3}{l-3}}, \end{aligned}$$

where in the last step we used the equality $\sum_{a=j+1}^{n} \binom{n-a}{l-3} = \sum_{a=0}^{n-j-1} \binom{a}{l-3} = \binom{n-j}{l-2}$.

We next consider the asymptotic setting $n \to \infty$. For this purpose, we define $Q(l) = \binom{n-j}{l-2}/\binom{n-3}{l-3}$. For $(i,j) = (1,2)$ we have $Q(l) = \binom{n-2}{l-2}/\binom{n-3}{l-3} = \frac{n-2}{l-2}$. The sum $\sum_{l=k+1}^{dn} \frac{n-2}{l-2}$ converges to $n \ln \frac{dn}{k}$ for $n \to \infty$. Further, $\lim_{n\to\infty} \sum_{k=cn+1}^{dn-1} n \ln \frac{dn}{k} = n\,(F(dn) - F(cn))$ for $F(x) := x \ln \frac{dn}{x} + x$. Hence,

$$\lim_{n\to\infty} p_{12} = \lim_{n\to\infty} \beta n \left(dn \ln \frac{dn}{dn} + dn - cn \ln \frac{dn}{cn} - cn \right) = c \left(d - c \ln \frac{d}{c} - c \right).$$

In the case $(i,j) = (2,3)$ it holds that $Q(l) = \binom{n-3}{l-2}/\binom{n-3}{l-3} = \frac{n-l}{l-2}$ and we have $\lim_{n\to\infty} \sum_{l=k+1}^{dn} \frac{n-l}{l-2} = n \ln \frac{dn}{k} - dn + k$. Let $F(x) := nx\left(\ln \frac{dn}{x} - d + 1\right) + \frac{x^2}{2}$. Again, by bounding the sum by the corresponding integral we obtain

$$\begin{aligned} &\lim_{n\to\infty} \sum_{k=cn+1}^{dn} n \ln \frac{dn}{k} - dn + k \\ &= F(dn) - F(cn) \\ &= dn^2 \left(\ln \frac{dn}{dn} - d + 1 \right) + \frac{d^2 n^2}{2} - cn^2 \left(\ln \frac{dn}{cn} - d + 1 \right) - \frac{c^2 n^2}{2} \\ &= n^2 \left(-d^2 + d + \frac{d^2}{2} - c \ln \frac{d}{c} + cd - c - \frac{c^2}{2} \right) \\ &= n^2 \left(d - c \ln \frac{d}{c} - c - \frac{d^2}{2} + cd - \frac{c^2}{2} \right). \end{aligned}$$

Multiplying the last term with $\lim_{n\to\infty} \beta = c/n^2$ gives the claim for p_{23}. ◀

Proof of Lemma 4.4. Suppose i is accepted first and j is accepted as the second item in the input sequence π. Consider the sequence π' obtained from π by swapping i with j. Since j and i are the first two elements beating the best sampling item in π', Algorithm 2 will select j and i on input π'. Hence, the number of permutations must be the same for both events, which implies the claim. ◀

Proof of Lemma 4.5. The argument is similar to the proof of Lemma 4.4. Consider any input sequence π where i is selected first and j second. We know that the best item a from sampling has profit $v_a < v_j < v_i$ and thus any item k with $i < k < j$ must occur after j. Let π' be the sequence obtained from π by swapping i with k. Now, i is behind k and j, thus Algorithm 2 will accept k and j. Again, this proves $p_{ij} = p_{kj}$ since the numbers of corresponding permutations are equal. ◀

The next lemma is used in the proof of Lemma 4.7 to show that for the given lists of parameters, we have $\beta_w \geq \gamma_w$.

▶ **Lemma A.1.** *Let $f(x) = 2\ln x - 6x + 2x^2 - \frac{x^3}{3}$. For parameters c,d with $f(c) \geq f(d)$ it holds that $\beta_w \geq \gamma_w$ where $2 \leq w \leq 5$.*

Proof. The function f is chosen in a way that $f(c) \geq f(d)$ is equivalent to $\beta_5 \geq \gamma_5$. This can be verified easily, using $\beta_5 = p_{\mathrm{D}} = p_1$, $\gamma_5 = p_{\mathrm{E}} + p_{\mathrm{F}} + p_{\mathrm{G}} = p_2 + p_3 + p_4$, and Lemma 4.2. Therefore, the claim for $w = 5$ holds by assumption. For $2 \leq w \leq 4$, the claims follow immediately from $f(c) \geq f(d)$ and the symmetry property of Lemma 4.4:

$$\beta_2 = p_{\mathrm{H}} = p_1 - p_{12} = p_1 - p_{21} \geq p_2 - p_{21} = p_{\mathrm{J}} = \gamma_2$$
$$\beta_3 = p_{\mathrm{I}} = p_1 - p_{13} = p_1 - p_{31} \geq p_2 + p_3 - p_{31} = p_{\mathrm{E}} + p_{\mathrm{L}} = \gamma_3$$
$$\beta_4 = p_{\mathrm{D}} + p_K = p_1 + p_2 - p_{23} \geq p_1 - p_{32} \geq p_3 - p_{32} = p_{\mathrm{M}} = \gamma_4\,. \qquad ◀$$

B Missing Proofs for the GAP Result

B.1 Large Options

Algorithm 4 Algorithm for edge-weighted bipartite matching from [26] (extended by our parameters c, d).

Input : Offline vertex set R, number of online vertices $n = |L|$,
parameters $c, d \in (0, 1)$ with $c < d$.
Output : Matching M.
Set $M = \emptyset$.
Let ℓ be the current round and l be the online vertex of round ℓ.
if $1 \leq \ell \leq cn$ **then**
  Sampling phase – do not add any edge.
if $cn + 1 \leq \ell \leq dn$ **then**
  Let $M^{(\ell)}$ be a maximum-weight matching for the graph in round ℓ.
  Let $e^{(\ell)} \in M^{(\ell)}$ be the edge incident to l.
  if $M \cup e^{(\ell)}$ *is a matching* **then**
    Add $e^{(\ell)}$ to M.
  if $\ell > dn$ **then**
    Do not add any edge.

Proof of Lemma 6.2. Let $e^{(\ell)}$ be the tentative edge of round ℓ and let $Q \subseteq L$ with $|Q| = \ell$ be the set of visible vertices from this round. Since each vertex from Q has the same probability of $1/\ell$ to arrive in round ℓ, we have

$$\mathbf{E}\left[w(e^{(\ell)})\right] = \sum_{e=\{l,r\}\in M^{(\ell)}} \mathbf{Pr}\left[l \text{ arrives in round } \ell\right] w(e) = \frac{1}{\ell} w(M^{(\ell)})\,. \tag{4}$$

Let $M^* = M^{(n)}$ be a maximum weight (offline) matching and $M^*_Q = \{e = \{l,r\} \in M^* \mid l \in Q\}$. We have $w(M^{(\ell)}) \geq w(M^*_Q)$, since $M^{(\ell)}$ is an optimal and M^*_Q a feasible matching for the graph revealed in round ℓ. As Q can be seen as uniformly drawn among all ℓ-element subsets, each vertex l has probability ℓ/n to be in Q. It follows

$$\mathbf{E}\left[w(M^{(\ell)})\right] \geq \mathbf{E}\left[w(M^*_Q)\right] = \sum_{e=\{l,r\}\in M^*} \mathbf{Pr}\left[l \in Q\right] w(e) = \frac{\ell}{n} w(M^*)\,. \tag{5}$$

Combining (4) and (5) concludes the proof. ◀

Proof of Lemma 6.3. In each round k, the vertex r can only be matched if it is incident to the tentative edge $e^{(k)} \in M^{(k)}$ of this round, i.e., $e^{(k)} = \{l,r\}$ where $l \in L$ is the online vertex of round k. As l can be seen as uniformly drawn among all k visible nodes (particularly, independent from the order of the previous $k-1$ items), l has probability $1/k$ to arrive in round k. Consequently, r is not matched in round k with probability $1 - 1/k$. This argument applies to all rounds $cn+1, \ldots, \ell$. Therefore,

$$\mathbf{Pr}\left[\xi(r,\ell)\right] \geq \prod_{k=cn+1}^{\ell} 1 - \frac{1}{k} = \prod_{k=cn+1}^{\ell} \frac{k-1}{k} = \frac{cn}{\ell}\,.$$ ◀

B.2 Small Options

For δ-small options we use the LP-based algorithm from [27, Sec. 3.3] and analyze it within our algorithmic framework. In order to make this paper self-contained, we give a linear program for GAP (LP 1), the algorithm, and its corresponding proofs.

$$\begin{aligned}
\text{maximize} \quad & \sum_{\substack{i\in I_S \\ r\in R}} v_{i,r} x_{i,r} & & \\
\text{subject to} \quad & \sum_{i\in I_S} s_{i,r} x_{i,r} \leq W_r & & \forall r \in R \\
& \sum_{r\in R} x_{i,r} \leq 1 & & \forall i \in I_S \\
& x_{i,r} \in \{0,1\} & & \forall (i,r) \in I_S \times R
\end{aligned} \tag{LP 1}$$

Let $\mathcal{A}_S$ be Algorithm 5. After a sampling phase of dn rounds, in each round ℓ the algorithm computes an optimal solution $x^{(\ell)}$ of the relaxation of LP 1 for $I_S(\ell)$. Here, $I_S(\ell)$ denotes the instance of small options revealed so far. Now, the decision to which resource the current online item i is assigned, if at all, is made by randomized rounding using $x^{(\ell)}$: Resource $r \in R$ is chosen with probability $x^{(\ell)}_{i,r}$ and the item stays unassigned with probability $1 - \sum_{r\in R} x^{(\ell)}_{i,r}$. Note that it is only feasible to assign the item to the chosen resource if its remaining capacity is at least δW_r.

Algorithm 5 GAP algorithm for small options from [27, Sec. 3.3].

Input : Random order sequence of small options,
parameter $d \in (0,1)$.
Output : Integral GAP assignment.
Let ℓ be the current round and i be the online item of round ℓ.
if $1 \leq \ell \leq dn$ **then**
  Sampling phase – do not assign any item.
if $dn+1 \leq \ell \leq n$ **then**
  Let $x^{(\ell)}$ be an optimal fractional solution of LP 1 for $I_S(\ell)$.
  Choose a resource r (possibly none), where r has probability $x^{(\ell)}_{i,r}$.
  if *the remaining capacity of* r *is at least* δW_r **then**
    Assign i to r.

To analyze Algorithm 5, we consider the gain of profit in round $\ell \geq dn+1$, denoted by A_ℓ. For this purpose, let $i^{(\ell)}$ be the item of that round and $r^{(\ell)}$ the resource chosen by the algorithm. Now, it holds that $\mathbf{E}[A_\ell] = \mathbf{E}\left[v_{i^{(\ell)},r^{(\ell)}}\right] \mathbf{Pr}\left[i^{(\ell)} \text{ can be assigned to } r^{(\ell)}\right]$, where in the first term, the expectation is over the item arriving in round ℓ and the resource chosen by the algorithm. The latter term only depends on the resource consumption of $r^{(\ell)}$ in earlier rounds. In the next two lemmas we give lower bounds for both terms.

▶ **Lemma B.1** ([27, Sec. 3.3]). *For any round $\ell \geq dn+1$, it holds that $\mathbf{E}\left[v_{i^{(\ell)},r^{(\ell)}}\right] \geq \frac{1}{n}\,\mathrm{OPT}_S$.*

Proof. The proof is similar to Lemma 6.2. As we consider a fixed round ℓ, we write i and r instead of $i^{(\ell)}$ and $r^{(\ell)}$ for ease of presentation. Further, we write $v(\alpha) := \sum_{j \in I_S} \sum_{s \in R} \alpha_{j,s} v_{j,s}$ for the profit of a fractional assignment α.

Fix any set Q of ℓ visible items in round ℓ. Let $x^{(n)}$ be an optimal (offline) solution to the relaxation of LP 1. Further, let $x^{(n)}|_Q$ denote the restriction of $x^{(n)}$ to the items in Q, i.e., $(x^{(n)}|_Q)_{j,s} = x^{(n)}_{j,s}$ if $j \in Q$ and $(x^{(n)}|_Q)_{j,s} = 0$ if $j \notin Q$. Since $x^{(n)}|_Q$ is a feasible and $x^{(\ell)}$ is an optimal solution for Q, we have $\mathbf{E}\left[v(x^{(\ell)})\right] \geq \mathbf{E}\left[v(x^{(n)}|_Q)\right]$. As in a random permutation each item has the same probability of ℓ/n to be in Q, it holds that

$$\mathbf{E}\left[v(x^{(\ell)})\right] \geq \mathbf{E}\left[v(x^{(n)}|_Q)\right] = \sum_{j \in I_S} \sum_{s \in R} \mathbf{Pr}[j \in Q]\, x^{(n)}_{j,s} v_{j,s} = \frac{\ell}{n} v(x^{(n)}) = \frac{\ell}{n}\,\mathrm{OPT}_S\,. \quad (6)$$

Similarly, each item from Q is the current online item i with probability $1/\ell$. The resource s, to which an item j gets assigned, is determined by randomized rounding using $x^{(\ell)}_{j,s}$. Therefore we get

$$\mathbf{E}[v_{i,r}] = \sum_{j \in Q} \sum_{s \in R} \mathbf{Pr}[j = i, s = r]\, v_{j,s} = \sum_{j \in Q} \sum_{s \in R} \frac{1}{\ell} x^{(\ell)}_{j,s} v_{j,s} = \frac{1}{\ell} v(x^{(\ell)})\,. \quad (7)$$

Combining (6) and (7) gives the claim. ◀

Hence, by the previous lemma the expected gain of profit in each round is a $(1/n)$-fraction of OPT_S, supposing the remaining resource capacity is large enough. The probability for the latter event is considered in the following lemma. Here, a crucial property is that we deal with δ-small options. Let $\Delta = \frac{1}{1-\delta}$.

▶ **Lemma B.2.** *For any round $\ell \geq dn+1$, we have $\mathbf{Pr}\left[i^{(\ell)} \text{ can be assigned to } r^{(\ell)}\right] \geq \frac{c}{d}\left(1 - \Delta \ln \frac{\ell}{dn}\right)$.*

Proof. Let ξ be the event that no item is assigned to r after round dn. Note that ξ does not necessarily hold, since $\mathcal{A}_L$ might already have assigned items to r in earlier rounds. By Lemma 6.3, $\mathbf{Pr}[\xi] \geq \frac{c}{d}$. Therefore, it remains to show $\mathbf{Pr}\left[i^{(\ell)} \text{ can be assigned to } r^{(\ell)} \mid \xi\right] \geq 1 - \Delta \ln \frac{\ell}{dn}$.

For this purpose, assume that ξ holds and let X denote the resource consumption of r after round $\ell - 1$. Further, let X_k be the resource consumption of r in round $k < \ell$. We have $X = \sum_{k=dn+1}^{\ell-1} X_k$. Let Q be the set of k visible items in round k. The set Q can be seen as uniformly drawn from all k-item subsets and any item $j \in Q$ is the current online item of round k with probability $1/k$. Now, the algorithm assigns any item j to resource r with probability $x_{j,r}^{(k)}$, thus

$$\mathbf{E}[X_k] = \sum_{j \in Q} \mathbf{Pr}[j \text{ occurs in round } k]\, s_{j,r} x_{j,r}^{(k)} = \frac{1}{k} \sum_{j \in Q} s_{j,r} x_{j,r}^{(k)} \leq \frac{W_r}{k}\,, \tag{8}$$

where the last inequality follows from the capacity constraint for resource r in LP 1. By linearity of expectation and inequality (8), the expected resource consumption up to round ℓ is thus

$$\mathbf{E}[X] = \sum_{k=dn+1}^{\ell-1} \mathbf{E}[X_k] \leq \sum_{k=dn+1}^{\ell-1} \frac{W_r}{k} \leq W_r \ln \frac{\ell}{dn}\,. \tag{9}$$

Now, since $i^{(\ell)}$ is δ-small, $X < (1-\delta)W_r$ implies $X + s_{i^{(\ell)},r^{(\ell)}} \leq W_r$ in which case the assignment is feasible. Using (9) and Markov's inequality, we obtain

$$\mathbf{Pr}[X < (1-\delta)W_r] = 1 - \mathbf{Pr}[X \geq (1-\delta)W_r] \geq 1 - \frac{\mathbf{E}[X]}{(1-\delta)W_r} \geq 1 - \Delta \ln \frac{\ell}{dn}\,. \qquad \blacktriangleleft$$

Now, the bound on the competitive ratio of $\mathcal{A}_S$ from Lemma 6.5 follows.

Proof of Lemma 6.5. We add the expected profits in single rounds using Lemmas B.1 and B.2.

$$\begin{aligned}
\mathbf{E}[\mathcal{A}_S] &= \sum_{\ell=dn+1}^{n} \mathbf{E}[A_\ell] = \sum_{\ell=dn+1}^{n} \mathbf{E}\left[v_{i^{(\ell)},r^{(\ell)}}\right] \mathbf{Pr}\left[i^{(\ell)} \text{ can be assigned to } r^{(\ell)}\right] \\
&\geq \sum_{\ell=dn+1}^{n} \frac{1}{n} \operatorname{OPT}_S \frac{c}{d} \left(1 - \Delta \ln \frac{\ell}{dn}\right) = \frac{c}{dn} \left(\sum_{\ell=dn+1}^{n} 1 - \Delta \ln \frac{\ell}{dn}\right) \operatorname{OPT}_S \\
&= \frac{c}{dn} \left(n - dn - \Delta \sum_{\ell=dn+1}^{n} \ln \frac{\ell}{dn}\right) \operatorname{OPT}_S\,.
\end{aligned}$$

Since $\frac{\ell}{dn}$ is monotone increasing in ℓ, we have $\sum_{\ell=dn+1}^{n} \ln \frac{\ell}{dn} \leq \int_{dn+1}^{n+1} \ln \frac{\ell}{dn}\, \mathrm{d}\ell$ and this integral evaluates to $(n+1)\ln\frac{n+1}{dn+1} - (n+1) - (dn+1)\ln\frac{dn+1}{dn} + (dn+1)$. For $n \to \infty$, this approaches $n \ln \frac{1}{d} - n + dn$. Hence, we have $\lim_{n\to\infty} \mathbf{E}[\mathcal{A}_S] \geq \frac{c}{d}\left((1+\Delta)(1-d) - \Delta \ln \frac{1}{d}\right) \operatorname{OPT}_S$. $\blacktriangleleft$

Appendix C

Best Fit Bin Packing with Random Order Revisited

Bibliographic information S. Albers, A. Khan, and L. Ladewig. Best fit bin packing with random order revisited. In *Proc. 45th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 170 of *LIPIcs*, pages 7:1–7:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

Summary In this paper, we revisit the BEST FIT algorithm for online bin packing in the random-order model. A seminal result by Kenyon [SODA'96] establishes lower and upper bounds of 1.08 and 1.5 on the asymptotic approximation ratio of BEST FIT under random arrival order, respectively. We make progress on this problem in several directions.

First, we consider the case where all items are larger than 1/3 of the bins' capacity. We show that the asymptotic approximation ratio is at most 5/4 and converges quickly to this value. For the proof, we introduce the notion of *good order pairs* and exploit a monotonicity property which holds only in this case. Second, we construct an improved lower bound of 1.10 on the asymptotic approximation ratio, improving upon the long-standing lower bound of 1.08. Finally, we initiate the study of *absolute* approximation ratio under random order. For the BEST FIT algorithm, we show that this value is 1.3 and thus, significantly larger than the currently best lower bound on the asymptotic approximation ratio.

Individual contributions

- Development of the main technical arguments in Section 3 (good order pairs, monotonicity) and corresponding proofs
- Construction and analysis of the asymptotic lower bound from Section 4
- Composition of the manuscript including all technical and non-technical parts (refined based on discussions with co-authors)

Best Fit Bin Packing with Random Order Revisited

Susanne Albers
Technische Universität München, Germany
albers@in.tum.de

Arindam Khan
Indian Institute of Science, Bangalore, India
arindamkhan@iisc.ac.in

Leon Ladewig
Technische Universität München, Germany
ladewig@in.tum.de

Abstract

Best Fit is a well known online algorithm for the bin packing problem, where a collection of one-dimensional items has to be packed into a minimum number of unit-sized bins. In a seminal work, Kenyon [SODA 1996] introduced the (asymptotic) *random order ratio* as an alternative performance measure for online algorithms. Here, an adversary specifies the items, but the order of arrival is drawn uniformly at random. Kenyon's result establishes lower and upper bounds of 1.08 and 1.5, respectively, for the random order ratio of Best Fit. Although this type of analysis model became increasingly popular in the field of online algorithms, no progress has been made for the Best Fit algorithm after the result of Kenyon.

We study the random order ratio of Best Fit and tighten the long-standing gap by establishing an improved lower bound of 1.10. For the case where all items are larger than 1/3, we show that the random order ratio converges quickly to 1.25. It is the existence of such large items that crucially determines the performance of Best Fit in the general case. Moreover, this case is closely related to the classical maximum-cardinality matching problem in the fully online model. As a side product, we show that Best Fit satisfies a monotonicity property on such instances, unlike in the general case.

In addition, we initiate the study of the *absolute* random order ratio for this problem. In contrast to asymptotic ratios, absolute ratios must hold even for instances that can be packed into a small number of bins. We show that the absolute random order ratio of Best Fit is at least 1.3. For the case where all items are larger than 1/3, we derive upper and lower bounds of 21/16 and 1.2, respectively.

2012 ACM Subject Classification Theory of computation → Online algorithms

Keywords and phrases Online bin packing, random arrival order, probabilistic analysis

Digital Object Identifier 10.4230/LIPIcs.MFCS.2020.7

Funding Work supported by the European Research Council, Grant Agreement No. 691672.

1 Introduction

One of the fundamental problems in combinatorial optimization is *bin packing*. Given a list $I = (x_1, \ldots, x_n)$ of n items with sizes from $(0, 1]$ and an infinite number of unit-sized bins, the goal is to pack all items into the minimum number of bins. Formally, a *packing* is an assignment of items to bins such that for any bin, the sum of assigned items is at most 1. While an offline algorithm has complete information about the items in advance, in the online variant, items are revealed one by one. Therefore, an online algorithm must pack x_i without knowing future items $x_{i+1}, \ldots, x_n$ and without modifying the packing of previous items.

As the problem is strongly NP-complete [15], research mainly focuses on efficient approximation algorithms. The offline problem is well understood and admits even approximation schemes [8, 18, 26]. The online variant is still a very active field in the community [6], as the asymptotic approximation ratio of the best online algorithm is still unknown [2, 3]. As one of the first algorithms for the problem, Garey et al. proposed the algorithms Best Fit and First Fit [14]. Johnson published the Next Fit algorithm briefly afterwards [22]. All of these algorithms work in the online setting and attract by their simplicity: Suppose that x_i is the current item to pack. The algorithms work as follows:

Best Fit (BF) Pack x_i into the fullest bin possible, open a new bin if necessary.

First Fit (FF) Maintain a list of bins ordered by the time at which they were opened. Pack x_i into the first possible bin in this list, open a new bin if necessary.

Next Fit (NF) Pack x_i into the bin opened most recently if possible; open a new bin if necessary.

Another important branch of online algorithms is based on the HARMONIC algorithm [29]. This approach has been massively tuned and generalized in a sequence of papers [2, 35, 36].

To measure the performance of an algorithm, different metrics exist. For an algorithm $\mathcal{A}$, let $\mathcal{A}(I)$ and $\mathrm{OPT}(I)$ denote the number of bins used by $\mathcal{A}$ and an optimal offline algorithm, respectively, to pack the items in I. Let $\mathcal{I}$ denote the set of all item lists. The most common metric for bin packing algorithms is the *asymptotic (approximation) ratio* defined as

$$R_{\mathcal{A}}^{\infty} = \limsup_{k\to\infty} \sup_{I\in\mathcal{I}} \{\mathcal{A}(I)/\,\mathrm{OPT}(I) \mid \mathrm{OPT}(I) = k\}\,.$$

Note that $R_{\mathcal{A}}^{\infty}$ focuses on instances where $\mathrm{OPT}(I)$ is large. This avoids anomalies typically occurring on lists that can be packed optimally into few bins. However, many bin packing algorithms are also studied in terms of the stronger *absolute (approximation) ratio*

$$R_{\mathcal{A}} = \sup_{I\in\mathcal{I}} \{\mathcal{A}(I)/\,\mathrm{OPT}(I)\}\,.$$

Here, the approximation ratio $R_{\mathcal{A}}$ must hold for each possible input. An online algorithm with (absolute or asymptotic) ratio α is also called *α-competitive*.

Table 1 shows the asymptotic and absolute approximation ratios of the three heuristics Best Fit, First Fit, and Next Fit. Interestingly, for these algorithms both metrics coincide. While the asymptotic ratios of Best Fit and Next Fit were established already in early work [23], the absolute ratios have been settled rather recently [9, 10].

Note that the above performance measures are clearly worst-case orientated. An adversary can choose items and present them in an order that forces the algorithm into its worst possible behavior. In the case of Best Fit, hardness examples are typically based on lists where small items occur before large items [14]. In contrast, it is known that Best Fit performs significantly better if items appear in non-increasing order [23]. For real-world instances, it seems overly pessimistic to assume adversarial order of input. Moreover, sometimes worst-case ratios hide interesting properties of algorithms that occur in average cases. This led to the development of alternative measures.

A natural approach that goes beyond worst-case was introduced by Kenyon [28] in 1996. In the model of random order arrivals, the adversary can still specify the items, but the arrival order is permuted randomly. The performance measure described in [28] is based on the asymptotic ratio, but can be applied to absolute ratios likewise. In the resulting performance metrics, an algorithm must satisfy its performance guarantee in expectation

Table 1 Approximation ratios in different metrics of common bin packing heuristics. In R_{NF}, the symbol γ refers to the total size of items in the list.

Algorithm $\mathcal{A}$	Abs. ratio $R_{\mathcal{A}}$	Asym. ratio $R^{\infty}_{\mathcal{A}}$	Asym. random order ratio $RR^{\infty}_{\mathcal{A}}$
Best Fit	1.7 [10]	1.7 [23]	$1.08 \leq RR^{\infty}_{BF} \leq 1.5$ [28]
First Fit	1.7 [9]	1.7 [23]	–
Next Fit	$2 - 1/\lceil\gamma\rceil$ [4]	2 [22]	2 [24]

over all permutations. We define

$$RR^{\infty}_{\mathcal{A}} = \limsup_{k\to\infty} \sup_{I\in\mathcal{I}} \{\mathrm{E}[\mathcal{A}(I^{\sigma})]/\,\mathrm{OPT}(I) \mid \mathrm{OPT}(I) = k\} \quad \text{and}$$
$$RR_{\mathcal{A}} = \sup_{I\in\mathcal{I}} \{\mathrm{E}[\mathcal{A}(I^{\sigma})]/\,\mathrm{OPT}(I)\}$$

as the *asymptotic random order ratio* and the *absolute random order ratio* of algorithm $\mathcal{A}$, respectively. 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.

1.1 Related work

The following literature review only covers results that are most relevant to our work. We refer the reader to the article [7] by Coffman et al. for an extensive survey on (online) bin packing. For further problems studied in the random order model, see [17].

Bin packing. Kenyon introduced the notion of asymptotic random order ratio $RR^{\infty}_{\mathcal{A}}$ for online bin packing algorithms in [28]. For the Best Fit algorithm, Kenyon proves an upper bound of 1.5 on $RR^{\infty}_{\mathrm{BF}}$, demonstrating that random order significantly improves upon $R^{\infty}_{\mathrm{BF}} = 1.7$. However, it is conjectured in [7, 28] that the actual random order ratio is close to 1.15. The proof of the upper bound crucially relies on the following scaling property: With high probability, the first t items of a random permutation can be packed optimally into $\frac{t}{n}\,\mathrm{OPT}(I) + o(n)$ bins. On the other side, Kenyon proves that $RR^{\infty}_{\mathrm{BF}} \geq 1.08$. This lower bound is obtained from the weaker i.i.d-model, where item sizes are drawn independently and identically distributed according to a fixed probability distribution.

Coffman et al. [24] analyzed next-fit in the random order model and showed that $RR^{\infty}_{\mathrm{NF}} = 2$, matching the asymptotic approximation ratio $RR^{\infty}_{\mathrm{NF}} = 2$ (see Table 1). Fischer and Röglin [12] obtained analogous results for worst-fit [22] and smart next-fit [34]. Therefore, all three algorithms fail to perform better in the random order model than in the adversarial model.

A natural property of bin packing algorithms is monotonicity, which holds if an algorithm never uses fewer bins to pack I' than for I, where I' is obtained from I by increasing item sizes. Murgolo [33] showed that next-fit is monotone, while Best Fit and First Fit are not monotone in general. The concept of monotonicity also arises in related optimization problems, such as scheduling [16] and bin covering [12].

Bin covering. The dual problem of bin packing is bin covering, where the goal is to cover as many bins as possible. A bin is covered if it receives items of total size at least 1. Here, a well-studied and natural algorithm is Dual Next Fit (DNF). In the adversarial setting, DNF has asymptotic ratio $R^{\infty}_{\mathrm{DNF}} = 1/2$ which is best possible for any online algorithm [5]. Under random arrival order, Christ et al. [5] showed that $RR^{\infty}_{\mathrm{DNF}} \leq 4/5$. This upper bound was

improved later by Fischer and Röglin [11] to $RR^{\infty}_{\mathrm{DNF}} \leq 2/3$. The same group of authors further showed that $RR^{\infty}_{\mathrm{DNF}} \geq 0.501$, i.e., DNF performs strictly better under random order than in the adversarial setting [12].

Matching. Online matching can be seen as the key problem in the field of online algorithms [32]. Inspired by the seminal work of Karp, Vazirani, and Vazirani [27], who introduced the online bipartite matching problem with one-sided arrivals, the problem has been studied in many generalizations. Extensions include fully online models [13, 19, 20], vertex-weighted versions [1, 21] and, most relevant to our work, random arrival order [21, 31].

1.2 Our results

While several natural algorithms fail to perform better in the random order model, Best Fit emerges as a strong candidate in this model. The existing gap between 1.08 and 1.5 clearly leaves room for improvement; closing (or even narrowing) this gap has been reported as challenging and interesting open problem in several papers [5, 17, 24].

To the best of our knowledge, our work provides the first new results on the problem since the seminal work by Kenyon. Below we describe our results in detail. In the following theorems, the expectation is over the permutation σ drawn uniformly at random.

Case of 1/3-large items

If all items are strictly larger than $1/3$, the objective is to maximize the number of bins containing two items. This problem is closely related to finding a maximum-cardinality matching in a vertex-weighted graph; our setting corresponds with the fully online model studied in [1] under random order arrival. Also in the analysis from [28], this special case arises. There, it is sufficient to argue that $\mathrm{BF}(I) \leq \frac{3}{2}\,\mathrm{OPT}(I) + 1$ under adversarial order. We show that Best Fit performs significantly better under random arrival order:

▶ **Theorem 1.1.** *For any list I of items larger than $1/3$, we have $\mathrm{E}[\mathrm{BF}(I^{\sigma})] \leq \frac{5}{4}\,\mathrm{OPT}(I) + \frac{1}{4}$.*

The proof of Theorem 1.1 is developed in Section 3 and based on several pillars. First, we show that Best Fit is monotone in this case (Proposition 3.2), unlike in the general case [33]. This property can be used to restrict the analysis to instances with well-structured optimal packing. The main technical ingredient is introduced in Section 3.3 with Lemma 3.5 as the key lemma. Here, we show that Best Fit maintains some parts of the optimal packing, depending on certain structures of the input sequence. We identify these structures and show that they occur with constant probability for a random permutation. It seems likely that this property can be used in a similar form to improve the bound $RR^{\infty}_{\mathrm{BF}} \leq 1.5$ for the general case: Under adversarial order, much hardness comes from relatively large items of size more than $1/3$; in fact, if all items have size at most $1/3$, an easy argument shows $4/3$-competitiveness even for adversarial arrival order [23].

Moreover, it is natural to ask for the performance in terms of absolute random order ratio. It is a surprising and rather recent result that for Best Fit, absolute and asymptotic ratios coincide. The result of [28] has vast additive terms and it seems that new techniques are required for insights into the absolute random order ratio. In Section 3.4, we show an upper bound of $21/16$ for $1/3$-large items, which is complemented by a lower bound of $6/5$.

▶ **Proposition 1.2.** *For any list I of items larger than $1/3$, we have $\mathrm{E}[\mathrm{BF}(I^{\sigma})] \leq \frac{21}{16}\,\mathrm{OPT}(I)$.*

▶ **Proposition 1.3.** *There is a list I of items larger than $1/3$ with $\mathrm{E}[\mathrm{BF}(I^{\sigma})] > \frac{6}{5}\,\mathrm{OPT}(I)$.*

A proof sketch of Proposition 1.3 is presented in Section 4.2.

Lower bounds

We also make progress on the hardness side, which is presented in Section 4. First, we show that the asymptotic random order ratio is larger than 1.10, improving the previous lower bound of 1.08 from [28].

▶ **Theorem 1.4.** *The asymptotic random order ratio of Best Fit is* $RR_{\mathrm{BF}}^{\infty} > 1.10$.

As it is typically challenging to obtain lower bounds in the random order model, we exploit the connection to the i.i.d-model. Here, items are drawn independently and identically distributed according to a fixed probability distribution. By defining an appropriate distribution, the problem can be analyzed using Markov chain techniques. Moreover, we present the first lower bound on the absolute random order ratio:

▶ **Theorem 1.5.** *The absolute random order ratio of Best Fit is* $RR_{\mathrm{BF}} \geq 1.30$.

Interestingly, our lower bound on the absolute random order ratio is notably larger than in the asymptotic case (see [28] and Theorem 1.4). This suggests either

- a significant discrepancy between RR_{BF} and $RR_{\mathrm{BF}}^{\infty}$, which is in contrast to the adversarial setting ($R_{\mathrm{BF}} = R_{\mathrm{BF}}^{\infty}$, see Table 1), or
- a disproof of the conjecture $RR_{\mathrm{BF}}^{\infty} \approx 1.15$ mentioned in [7, 28].

2 Notation

We consider a list $I = (x_1, \ldots, x_n)$ of n items throughout the paper. Due to the online setting, I is revealed in *rounds* $1, \ldots, n$. In round t, item x_t arrives and in total, the prefix list $I(t) := (x_1, \ldots, x_t)$ is revealed to the algorithm. The items in $I(t)$ are called the *visible* items of round t. We use the symbol x_t for the item itself and its size $x_t \in (0, 1]$ interchangeably. An item x_t is called *large* (L) if $x_t > 1/2$, *medium* (M) if $x_t \in \left(\frac{1}{3}, \frac{1}{2}\right]$, and *small* (S) if $x_t \leq 1/3$. We also say that x_t is α-large if $x_t > \alpha$.

Bins contain items and therefore can be represented as sets. As a bin usually can receive further items in later rounds, the following terms refer always to a fixed round. We define the *load* of a bin $\mathcal{B}$ as $\sum_{x_i \in \mathcal{B}} x_i$. Sometimes, we classify bins by their internal structure. We say $\mathcal{B}$ is of *configuration LM* (or $\mathcal{B}$ is an *LM-bin*) if it contains one large and one medium item. The configurations L, MM, etc. are defined analogously. Moreover, we call $\mathcal{B}$ a k-bin if it contains exactly k items. If a bin cannot receive further items in the future, it is called *closed*; otherwise, it is called *open*.

The number of bins which Best Fit uses to pack a list I is denoted by $\mathrm{BF}(I)$. We slightly abuse the notation and refer to the corresponding packing by $\mathrm{BF}(I)$ as well whenever the exact meaning is clear from the context. Similarly, we denote by $\mathrm{OPT}(I)$ the number of bins and the corresponding packing of an optimal offline solution.

Finally, for any natural number n we define $[n] := \{1, \ldots, n\}$. Let $\mathcal{S}_n$ be the set of permutations in $[n]$. If not stated otherwise, σ refers to a permutation drawn uniformly at random from $\mathcal{S}_n$.

3 Upper bound for 1/3-large items

In this section, we consider the case where I contains no small items, i.e., where all items are 1/3-large. In Sections 3.1 to 3.3 we develop the technical foundations. The final proofs of Theorem 1.1 and Proposition 1.2 are presented in Section 3.4.

3.1 Monotonicity

We first define the notion of monotone algorithms.

▶ **Definition 3.1.** *We call an algorithm monotone if increasing the size of one or more items cannot decrease the number of bins used by the algorithm.*

One might suspect that any reasonable algorithm is monotone. While this property holds for an optimal offline algorithm and some online algorithms as ext-fit [25], Best Fit is not monotone in general [33]. As a counterexample, consider the lists

$$I = (0.36,\ 0.65,\ \mathbf{0.34},\ 0.38,\ 0.28,\ 0.35,\ 0.62) \text{ and}$$
$$I' = (0.36,\ 0.65,\ \mathbf{0.36},\ 0.38,\ 0.28,\ 0.35,\ 0.62)\,.$$

Before arrival of the fifth item, $\mathrm{BF}(I(4))$ uses two bins $\{0.36, 0.38\}$ and $\{0.65, 0.34\}$, while $\mathrm{BF}(I'(4))$ uses three bins $\{0.36, 0.36\}$, $\{0.65\}$, and $\{0.38\}$. Now, the last three items fill up the existing bins in $\mathrm{BF}(I'(4))$ exactly. In contrast, these items open two further bins in the packing of $\mathrm{BF}(I(4))$. Therefore, $\mathrm{BF}(I) = 4 > 3 = \mathrm{BF}(I')$.

However, we can show that Best Fit is monotone for the case of 1/3-large items. Interestingly, 1/3 seems to be the threshold for the monotonicity of Best Fit: As shown in the counterexample from the beginning of this section, it is sufficient to have one item $x \in \left(\frac{1}{4}, \frac{1}{3}\right]$ to force Best Fit into anomalous behavior.

▶ **Proposition 3.2.** *Given a list I of items larger than 1/3 and a list I' obtained from I by increasing the sizes of one or more items, we have $\mathrm{BF}(I) \leq \mathrm{BF}(I')$.*

Sketch of proof. For simplicity, first assume that both lists differ only in the i-th element. All bins in any packing of I or I' contain at most two items. We call two 1-bins of $\mathrm{BF}(I)$ and $\mathrm{BF}(I')$ *pairwise-identical* if they contain items of the same size. Moreover, we call any two 2-bins of $\mathrm{BF}(I)$ and $\mathrm{BF}(I')$ *pairwise-closed*, as neither of the two bins can receive a further item. For ease of notation, let $I_t = I(t)$ and $I'_t = I'(t)$. We can show that at any time t, the packings $\mathrm{BF}(I_t)$ and $\mathrm{BF}(I'_t)$ are related in one of three ways:

(1) All bins are pairwise-identical or pairwise-closed.

(2) All bins are pairwise-identical or pairwise-closed, except for two 1-bins $B = \{b\}$ and $B' = \{b'\}$ in $\mathrm{BF}(I_t)$ and $\mathrm{BF}(I'_t)$, respectively, where $b < b'$.

(3) All bins are pairwise-identical or pairwise-closed, except for a 2-bin $C = \{c_1, c_2\}$ in $\mathrm{BF}(I_t)$ which does not exist in $\mathrm{BF}(I'_t)$, and two 1-bins $B'_1 = \{b'_1\}$, $B'_2 = \{b'_2\}$ in $\mathrm{BF}(I'_t)$ which do not exist in $\mathrm{BF}(I_t)$.

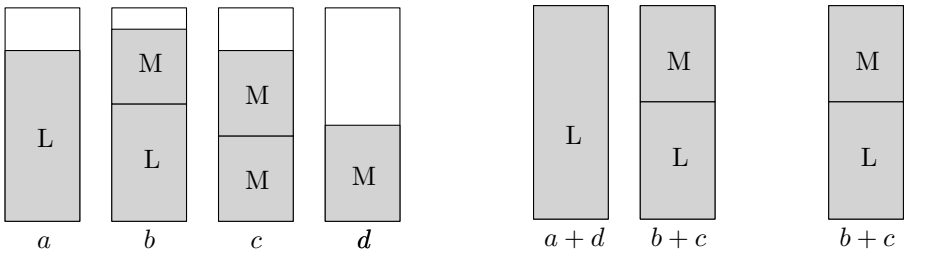
Note in all three cases, $\mathrm{BF}(I_t) \leq \mathrm{BF}(I'_t)$. As this property is maintained until $t = n$, it implies the lemma. ◀

The entire proof of Proposition 3.2 will be given in the full version of this paper.

3.2 Simplifying the instance

Let I be a list of items larger than 1/3. Note that both the optimal and the Best Fit packing use only bins of configurations L, LM, MM, and possibly one M-bin. However, we can assume a simpler structure without substantial implications on the competitiveness of Best Fit.

▶ **Lemma 3.3.** *Let I be any list that can be packed optimally into $\mathrm{OPT}(I)$ LM-bins. If Best Fit has (asymptotic or absolute) approximation ratio α for I, then it has (asymptotic or absolute) approximation ratio α for any list of items larger than 1/3 as well.*



(a) $\mathrm{OPT}(I_0) = \mathrm{OPT}(I_1)$. **(b)** $\mathrm{OPT}(I_2)$. **(c)** $\mathrm{OPT}(I_3)$.

Figure 1 Construction from Lemma 3.3 to eliminate L-, MM-, and M-bins in the optimal packing.

Proof. Let I_0 be a list of items larger than $1/3$ and let a, b, c, and $d \leq 1$ be the number of bins in $\mathrm{OPT}(I_0)$ with configurations L, LM, MM, and M, respectively (see Figure 1a). In several steps, we eliminate L-,MM-, and M-bins from $\mathrm{OPT}(I_0)$ while making the instance only harder for Best Fit.

First, we obtain I_1 from I_0 by replacing items of size $1/2$ by items of size $1/2 - \varepsilon$. By choosing $\varepsilon > 0$ small enough, i.e., $\varepsilon < \min\{\delta^+ - 1/2, 1/2 - \delta^-\}$, where $\delta^+ = \min\{x_i \mid x_i > 1/2\}$ and $\delta^- = \max\{x_i \mid x_i < 1/2\}$, it is ensured that Best Fit packs all items in the same bins than before the modification. Further, the modification does not decrease the number of bins in an optimal packing, so we have $\mathrm{BF}(I_0) = \mathrm{BF}(I_1)$ and $\mathrm{OPT}(I_0) = \mathrm{OPT}(I_1)$. Now, we obtain I_2 from I_1 by increasing item sizes: We replace each of the $a+d$ items packed in 1-bins in $\mathrm{OPT}(I_1)$ by large items of size 1. Moreover, any 2-bin (MM or LM) in $\mathrm{OPT}(I_1)$ contains at least one item smaller than $1/2$. These items are enlarged such that they fill their respective bin completely. Therefore, $\mathrm{OPT}(I_2)$ has $a + d$ L-bins and $b + c$ LM-bins (see Figure 1b). We have $\mathrm{OPT}(I_2) = \mathrm{OPT}(I_1)$ and, by Proposition 3.2, $\mathrm{BF}(I_2) \geq \mathrm{BF}(I_1)$. Finally, we obtain I_3 from I_2 by deleting the $a + d$ items of size 1. As size-1 items are packed separately in any feasible packing, $\mathrm{OPT}(I_3) = \mathrm{OPT}(I_2) - (a + d)$ and $\mathrm{BF}(I_3) = \mathrm{BF}(I_2) - (a + d)$. Note that $\mathrm{OPT}(I_3)$ contains only LM-bins (see Figure 1c) and, by assumption, Best Fit has (asymptotic or absolute) approximation ratio α for such lists. Therefore, in general we have a factor $\alpha \geq 1$ and an additive term β such that $\mathrm{BF}(I_3) \leq \alpha\,\mathrm{OPT}(I_3) + \beta$. It follows that

$$\mathrm{BF}(I_0) \leq \mathrm{BF}(I_2) = \mathrm{BF}(I_3) + (a + d) \leq \alpha\,\mathrm{OPT}(I_3) + (a + d) + \beta \leq \alpha\,\mathrm{OPT}(I_0) + \beta\,. \quad \blacktriangleleft$$

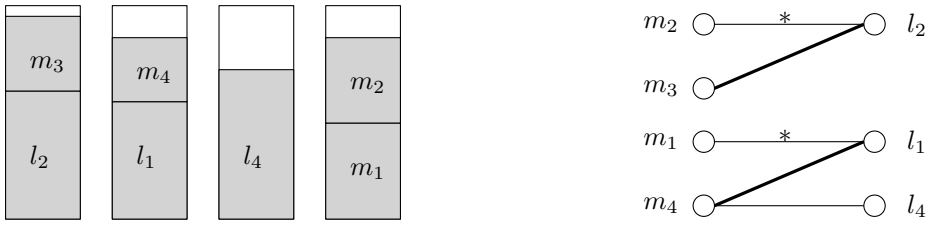
By Lemma 3.3, we can impose the following constraints on I without loss of generality.

Assumption. For the remainder of the section, we assume that the optimal packing of I has $k = \mathrm{OPT}(I)$ LM-bins. For $i \in [k]$, let l_i and m_i denote the large item and the medium item in the i-th bin, respectively. We call $\{l_i, m_i\}$ an *LM-pair*.

3.3 Good order pairs

If the adversary could control the order of items, he would send all medium items first, followed by all large items. This way, Best Fit opens $k/2$ MM-bins and k L-bins and therefore is 1.5-competitive. In a random permutation, we can identify structures with a positive impact on the Best Fit packing. This is formalized in the following random event.

▶ **Definition 3.4.** *Consider a fixed permutation $\pi \in \mathcal{S}_n$. We say that an LM-pair $\{l_i, m_i\}$ arrives in good order (or is a good order pair) if l_i arrives before m_i in π.*



(a) Best Fit packing BF($I(7)$). **(b)** Graph G_7.

Figure 2 Visualization of Example 3.6. In Figure 2b, BF-edges are solid, while OPT-edges are thin. An asterisk indicates an OPT-edge in good order.

Note that in the adversarial setting no LM-pair arrives in good order, while in a random permutation, this holds for any LM-pair independently with probability 1/2. The next lemma is central for the proof of Theorem 1.1. It shows that the number of LM-pairs in good order bound the number of LM-bins in the final Best Fit packing from below.

▶ **Lemma 3.5.** *Let $\pi \in \mathcal{S}_n$ be any permutation and let X be the number of LM-pairs arriving in good order in I^π. The packing* BF(I^π) *has at least X LM-bins.*

To prove Lemma 3.5, we model the Best Fit packing by the following bipartite graph: Let $G_t = (\mathcal{M}_t \cup \mathcal{L}_t, E_t^{\mathrm{BF}} \cup E_t^{\mathrm{OPT}})$, where $\mathcal{M}_t$ and $\mathcal{L}_t$ are the sets of medium and large items in $I^\pi(t)$, respectively. The sets of edges represent the LM-matchings in the Best Fit packing and in the optimal packing at time t, i.e.,

$$E_t^{\mathrm{BF}} = \left\{\{m,l\} \in (\mathcal{M}_t \times \mathcal{L}_t) \mid m \text{ and } l \text{ are packed into the same bin in BF}(I^\pi(t))\right\}$$
$$E_t^{\mathrm{OPT}} = \left\{\{m_i,l_i\} \in (\mathcal{M}_t \times \mathcal{L}_t) \mid i \in [k]\right\}.$$

We distinguish OPT-edges in good and bad order, according to the corresponding LM-pair. Note that G_t is not necessarily connected and may contain parallel edges. We illustrate the graph representation by a small example.

▶ **Example 3.6** (see Figure 2)**.** Let $\varepsilon > 0$ be sufficiently small and define for $i \in [4]$ large items $l_i = 1/2 + i\varepsilon$ and medium items $m_i = 1/2 - i\varepsilon$. Consider the list $I^\pi = (l_2, l_1, m_3, m_4, l_4, m_1, m_2, l_3)$. Figures 2a and 2b show the Best Fit packing and the corresponding graph G_7 before arrival of the last item. Note that I^π has two good order pairs ($\{l_1, m_1\}$ and $\{l_2, m_2\}$) and, according to Lemma 3.5, the packing has two LM-bins.

The proof of Lemma 3.5 essentially boils down to the following claim:

▷ Claim 3.7. In each round t and in each connected component C of G_t, the number of BF-edges in C is at least the number of OPT-edges in good order in C.

We first show how Lemma 3.5 follows from Claim 3.7. Then, we work towards the proof of Claim 3.7.

Proof of Lemma 3.5. Claim 3.7 implies that in G_n, the total number of BF-edges (summed over all connected components) is at least X. Therefore, the packing has at least X LM-bins and thus not less than the number of good order pairs X. ◀

Before proving Claim 3.7, it is reasonable to observe the following property of G_t.

▷ Claim 3.8. Consider the graph G_t for some $t \in [n]$. Let $Q = (b_w, a_{w-1}, b_{w-1}, \ldots, a_1, b_1)$ with $w \geq 1$ be a maximal alternating path such that $\{a_j, b_j\}$ is an OPT-edge in good order and $\{a_j, b_{j+1}\}$ is a BF-edge for any $j \in [w-1]$ (i.e., a-items and b-items represent medium and large items, respectively). It holds that $b_w \geq b_1$.

Proof. We show the claim by induction on w. Note that the items' indices only reflect the position along the path, not the arrival order. For $w = 1$, we have $Q = (b_w) = (b_1)$ and thus, the claim holds trivially.

Now, fix $w \geq 2$ and suppose that the claim holds for all paths Q' with $w' \leq w - 1$. We next prove $b_w \geq b_1$. Let $t' \leq t$ be the arrival time of the a-item a_d that arrived latest among all a-items in Q. We consider the graph $G_{t'-1}$, i.e., the graph immediately before arrival of a_d and its incident edges. Note that in $G_{t'-1}$, all items a_i with $i \in [w-1] \setminus \{d\}$ and b_i with $i \in [w-1]$ are visible. Let $Q' = (b_w, \ldots, a_{d+1}, b_{d+1})$ and $Q'' = (b_d, \ldots, a_1, b_1)$ be the connected components of b_w and b_1 in $G_{t'-1}$. As Q' and Q'' are maximal alternating paths shorter than Q, we obtain from the induction hypothesis $b_w \geq b_{d+1}$ and $b_d \geq b_1$. Note that b_{d+1} and b_1 were visible and packed into L-bins on arrival of a_d. Further, a_d and b_1 would fit together, as $a_d + b_1 \leq a_d + b_d \leq 1$. However, Best Fit packed a_d with b_{d+1}, implying $b_{d+1} \geq b_1$. Combining the inequalities yields $b_w \geq b_{d+1} \geq b_1$, which concludes the proof. ◁

Now, we are able to prove the remaining technical claim.

Proof of Claim 3.7. Note that the number of OPT-edges in good order can only increase on arrival of a medium item m_i where $\{m_i, l_i\}$ is an LM-pair in good order. Therefore, it is sufficient to verify Claim 3.7 in rounds $t_1 < \ldots < t_j$ such that in round t_i, item m_i arrives and l_i arrived previously.

Induction base. In round t_1, there is one OPT-edge $\{m_1, l_1\}$ in good order. We need to show that there exists at least one BF-edge in G_{t_1}, or, alternatively, at least one LM-bin in the packing. If the bin of l_1 contains a medium item different from m_1, we identified one LM-bin. Otherwise, Best Fit packs m_1 together with l_1 or some other large item, again creating an LM-bin.

Induction hypothesis. Fix $i \geq 2$ and assume that Claim 3.7 holds up to round t_{i-1}.

Induction step. We only consider the connected component of m_i, as by the induction hypothesis, the claim holds for all remaining connected components. If m_i is packed into an LM-bin, the number of BF-edges increases by one and the claim holds for round t_i. Therefore, assume that m_i is packed by Best Fit in an M- or MM-bin. This means that in G_{t_i}, vertex m_i is incident to an OPT-edge in good order, but not incident to any BF-edge. Let $P = (m_i, l_i, \ldots, v)$ be the maximal path starting from m_i alternating between OPT-edges and BF-edges.

Case 1: v is a medium item For illustration, consider Figure 2b with $m_i = m_2$ and $v = m_3$. Since P begins with an OPT-edge and ends with a BF-edge, the number of BF-edges in P equals the number of OPT-edges in P. The latter number is clearly at least the number of OPT-edges in good order in P.

Case 2: v is a large item For illustration, consider Figure 2b with $m_i = m_1$ and $v = l_4$. We consider two cases. If P contains at least one OPT-edge which is not in good order, the claim follows for the same argument as in Case 1.

Now, suppose that all OPT-edges in P are in good order. Let P' be the path obtained from P by removing the item m_i. As P' satisfies the premises of Claim 3.8, we obtain $l_i \geq v$. This implies that m_i and v would fit together, as $m_i + v \leq m_i + l_i \leq 1$. However, m_i is packed in an M- or MM-bin by assumption, although v is a feasible option on arrival of m_i. As this contradicts the Best Fit rule, we conclude that case 2 cannot happen. ◁

3.4 Final proofs

Finally, we prove the main result of this section.

Proof of Theorem 1.1. Let X be the number of good order pairs in I^σ and let Y be the number of LM-bins in the packing $\mathrm{BF}(I^\sigma)$. We have $Y \geq X$ by Lemma 3.5. For the remaining large and medium items, Best Fit uses $(k - Y)$ L-bins and $\lceil (k-Y)/2 \rceil$ MM-bins (including possibly one M-bin), respectively. Therefore,

$$\mathrm{BF}(I^\sigma) = Y + (k - Y) + \left\lceil \frac{k-Y}{2} \right\rceil \leq k + \left\lceil \frac{k-X}{2} \right\rceil = \frac{3k}{2} - \frac{X}{2} + \frac{\xi(X)}{2}\,, \tag{1}$$

where $\xi(X) = (k - X) \bmod 2$. Using linearity and monotonicity of expectation, we obtain

$$\mathrm{E}[\mathrm{BF}(I^\sigma)] \leq \frac{3k}{2} - \frac{\mathrm{E}[X]}{2} + \frac{\Pr[\xi(X) = 1]}{2}\,. \tag{2}$$

Since σ is uniformly distributed on $\mathcal{S}_n$, each LM-pair arrives in good order with probability $1/2$, independently of all other pairs. Therefore, X follows a binomial distribution with parameters k and $1/2$, implying $\mathrm{E}[X] = k/2$ and $\Pr[\xi(X) = 1] = 1/2$. Hence,

$$\mathrm{E}[\mathrm{BF}(I^\sigma)] \leq \frac{3k}{2} - \frac{k/2}{2} + \frac{1/2}{2} = \frac{5k}{4} + \frac{1}{4} = \frac{5}{4}\,\mathrm{OPT}(I) + \frac{1}{4}\,, \tag{3}$$

where we used $k = \mathrm{OPT}(I)$. This concludes the proof. ◀

To obtain a slightly weaker bound on the absolute random order ratio (Proposition 1.2), we analyze some special cases more carefully.

Proof of Proposition 1.2. For $k \geq 4$ the claim follows immediately from Equation (3):

$$\frac{\mathrm{E}[\mathrm{BF}(I^\pi)]}{\mathrm{OPT}(I)} = \frac{(5k)/4 + 1/4}{k} = \frac{5}{4} + \frac{1}{4k} \leq \frac{21}{16}\,.$$

Since Best Fit is clearly optimal for $k = 1$, it remains to verify the cases $k \in \{2, 3\}$.

$k = 2$ It is easily verified that there are 16 out of $4! = 24$ permutations where Best Fit is optimal and that it opens at most 3 bins otherwise. Therefore,

$$\mathrm{E}[\mathrm{BF}(I^\sigma)] = \frac{1}{4!} \cdot \left(16\,\mathrm{OPT}(I) + 8 \cdot \frac{3}{2}\,\mathrm{OPT}(I)\right) = \frac{7}{6}\,\mathrm{OPT}(I) < \frac{21}{16}\,\mathrm{OPT}(I)\,.$$

$k = 3$ When k is odd, there must be at least one LM-bin in the Best Fit packing: Suppose for contradiction that all M-items are packed in MM- or M-bins. As k is odd, there must be an item m_i packed in an M-bin. If l_i arrives before m_i, item l_i is packed in an L-bin, as there is no LM-bin. Therefore, Best Fit would pack m_i with l_i or some other L-item instead of opening a new bin. If l_i arrives after m_i, Best Fit would pack l_i with m_i or some other M-item. We have a contradiction in both cases.
Therefore, for $k = 3$ we have at least one LM-bin, even if no LM-pair arrives in good order. Consider the proof of Theorem 1.1. Instead of $Y \geq X$, we can use the stronger bound

$Y \geq X'$ with $X' := \max\{1, X\}$ on the number of LM-bins. The new random variable satisfies $\mathrm{E}[X'] = k/2 + 1/2^k$ and $\Pr[\xi(X') = 1] = 1/2 - 1/2^k$. Adapting Equations (1) and (2) appropriately, we obtain

$$\frac{\mathrm{E}[\mathrm{BF}(I^\sigma)]}{\mathrm{OPT}(I)} = \frac{1}{k} \cdot \left(\frac{3k}{2} - \frac{k/2 + 1/2^k}{2} + \frac{1/2 - 1/2^k}{2} \right) = \frac{5}{4} + \frac{1}{4k} - \frac{1}{k2^k} = \frac{31}{24} < \frac{21}{16} . \blacktriangleleft$$

4 Lower bounds

In this section, we present the improved lower bound on $RR^{\infty}_{\mathrm{BF}}$ (Theorem 1.4) and the first lower bound on the absolute random order ratio RR_{BF}.

4.1 Asymptotic random order ratio

Consider the i.i.d.-model, where the input is a sequence of independent and identically distributed (i.i.d) random variables. Here, the performance measure for an algorithm $\mathcal{A}$ is $\mathrm{E}[\mathcal{A}(I_n(F))]/\mathrm{E}[\mathrm{OPT}(I_n(F))]$, where $I_n(F) := (X_1, \ldots, X_n)$ is a list of n random variables drawn i.i.d according to F. This model is in general weaker than the random order model, which is why lower bounds in the random order model can be obtained from the i.i.d. model. This is formalized in the following lemma.

▶ **Lemma 4.1.** *Consider any online bin packing algorithm $\mathcal{A}$. Let F be a discrete distribution and $I_n(F) = (X_1, \ldots, X_n)$ be a list of i.i.d. samples. There exists a list I of n items such that for $n \to \infty$,*

$$\frac{\mathrm{E}[\mathcal{A}(I^\sigma)]}{\mathrm{OPT}(I)} \geq \frac{\mathrm{E}[\mathcal{A}(I_n(F))]}{\mathrm{E}[\mathrm{OPT}(I_n(F))]} .$$

Moreover, if there exists a constant $c > 0$ such that $X_i \geq c$ for all $i \in [n]$, we have $\mathrm{OPT}(I) \geq cn$.

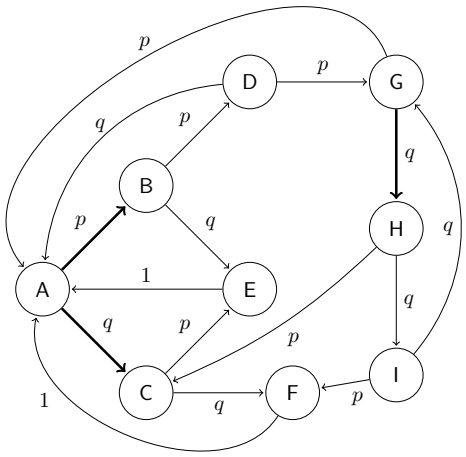
This technique has already been used in [28] to establish the previous bound of 1.08, however, without a formal proof. Apparently, the only published proofs of this reduction technique address bin covering [5, 11]. We will provide a constructive proof of Lemma 4.1 in the full version of this paper. Theorem 1.4 follows by combining Lemmas 4.1 and 4.2.

▶ **Lemma 4.2.** *There exists a discrete distribution F such that for $n \to \infty$, we have $\mathrm{E}[\mathcal{A}(I_n(F))] > \frac{11}{10} \mathrm{E}[\mathrm{OPT}(I_n(F))]$ and each sample X_i satisfies $X_i \geq 1/4$.*

Proof. Let F be the discrete distribution which gives an item of size $1/4$ with probability p and an item of size $1/3$ with probability $q := 1 - p$. First, we analyze the optimal packing. Let N_4 and N_3 be the number of items with size $1/4$ and $1/3$ in $I_n(F)$, respectively. We have

$$\mathrm{E}[\mathrm{OPT}(I_n(F))] \leq \mathrm{E}\left[\frac{N_4}{4} + \frac{N_3}{3} + 2 \right] = \frac{np}{4} + \frac{nq}{3} + 2 = n\left(\frac{1}{3} - \frac{p}{12} + \frac{2}{n} \right) .$$

Now, we analyze the expected behavior of Best Fit for $I_n(F)$. As the only possible item sizes are $1/4$ and $1/3$, we can consider each bin of load more than $3/4$ as closed. Moreover, the number of possible loads for open bins is small and Best Fit maintains at most two open bins at any time. Therefore, we can model the Best Fit packing by a Markov chain as follows. Let the nine states $\mathsf{A}, \mathsf{B}, \ldots, \mathsf{I}$ be defined as in Figure 3b. The corresponding



(a) Transition diagram.

State	Load of open bin(s)
A	–
B	1/4
C	1/3
D	2/4
E	7/12
F	2/3
G	3/4
H	3/4, 1/3
I	3/4, 2/3

(b) Description of states.

Figure 3 Markov chain from Lemma 4.2. Bold arcs in Figure 3a indicate transitions where Best Fit opens a new bin.

transition diagram is depicted in Figure 3a. This Markov chain converges to the stationary distribution

$$\omega = (\omega_{\mathsf{A}}, \dots, \omega_{\mathsf{I}}) = \frac{1}{\lambda}\left(1,\ p,\ q+pq\vartheta,\ p^2,\ 2pq+p^2q\vartheta,\ q^2+2pq^2\vartheta,\ \vartheta,\ q\vartheta,\ q^2\vartheta\right), \tag{4}$$

where we defined $\vartheta = \frac{p^3}{1-q^3}$ and $\lambda = \vartheta q\left(3-q^2\right)+\vartheta+3$. A formal proof of this fact will appear in the full version of this paper.

Let $V_{\mathsf{S}}(t)$ denote the number of visits to state $\mathsf{S} \in \{\mathsf{A}, \dots, \mathsf{I}\}$ up to time t. By a basic result from the theory of ergodic Markov chains (see [30, Sec. 4.7]), it holds that $\lim_{t\to\infty} V_{\mathsf{S}}(t) = n\omega_S$. In other words, the proportion of time spent in state S approaches its probability ω_{S} in the stationary distribution. This fact can be used to bound the total number of opened bins over time. Note that Best Fit opens a new bin on the transitions $A \to B$, $A \to C$, and $G \to H$ (see Figure 3a). Hence, $\mathrm{E}[\mathrm{BF}(I_n(F))] = n\left(\omega_{\mathsf{A}} + q\omega_{\mathsf{G}}\right)$. Setting $p = 0.60$, we obtain finally

$$\lim_{n\to\infty}\frac{\mathrm{E}[\mathrm{BF}(I_n(F))]}{\mathrm{E}[\mathrm{OPT}(I_n(F))]} \geq \lim_{n\to\infty}\frac{\omega_{\mathsf{A}}+q\omega_{\mathsf{G}}}{\frac{1}{3}-\frac{p}{12}+\frac{2}{n}} = \frac{1+q\vartheta}{\lambda\cdot\left(\frac{1}{3}-\frac{p}{12}\right)} > \frac{11}{10}\,. \qquad \blacktriangleleft$$

4.2 Absolute random order ratio

Theorem 1.5 follows from the following lemma.

▶ **Lemma 4.3.** *There exists a list I such that* $\mathrm{E}[\mathrm{BF}(I^\sigma)] = \frac{13}{10}\,\mathrm{OPT}(I)$.

Proof. Let $\varepsilon > 0$ be sufficiently small and let $I := (a_1, a_2, b_1, b_2, c)$ where

$$a_1 = a_2 = \frac{1}{3}+4\varepsilon, \quad b_1 = b_2 = \frac{1}{3}+16\varepsilon, \quad c = \frac{1}{3}-8\varepsilon\,.$$

An optimal packing of I needs two bins $\{a_1, a_2, c\}$ and $\{b_1, b_2\}$, thus $\mathrm{OPT}(I) = 2$. Best Fit needs two or three bins depending on the order of arrival.

Let E be the event that exactly one b-item arrives within the first two rounds. After the second item, the first bin is closed, as its load is at least $\frac{1}{3} + 16\varepsilon + \frac{1}{3} - 8\varepsilon = \frac{2}{3} + 8\varepsilon$. Among the remaining three items, there is a b-item of size $\frac{1}{3} + 16\varepsilon$ and at least one a-item of size $\frac{1}{3} + 4\varepsilon$. This implies that a third bin needs to be opened for the last item. As there are exactly $2 \cdot 3 \cdot 2! \cdot 3! = 72$ permutations where E happens, we have $\Pr[E] = \frac{72}{5!} = \frac{3}{5}$.

On the other side, Best Fit needs only two bins if one of the events F and G, defined in the following, happen. Let F be the event that both b-items arrive in the first two rounds. Then, the remaining three items fit into one additional bin. Moreover, let G be the event that the set of the first two items is a subset of $\{a_1, a_2, c\}$. Then, the first bin has load at least $\frac{2}{3} - 4\varepsilon$, thus no b-item can be packed there. Again, this ensures a packing into two bins. By counting permutations, we obtain $\Pr[F] = \frac{2! \cdot 3!}{5!} = \frac{1}{10}$ and $\Pr[G] = \frac{3 \cdot 2! \cdot 3!}{5!} = \frac{3}{10}$.

As the events E, F, and G partition the probability space, we obtain

$$\frac{\mathrm{E}[\mathrm{BF}(I^\sigma)]}{\mathrm{OPT}(I)} = \frac{\Pr[E] \cdot 3 + (\Pr[F] + \Pr[G]) \cdot 2}{2} = \frac{\frac{3}{5} \cdot 3 + \left(\frac{1}{10} + \frac{3}{10}\right) \cdot 2}{2} = \frac{13}{10}\,. \qquad \blacktriangleleft$$

The construction from the above proof is used in [23] to prove that Best Fit is 1.5-competitive under adversarial arrival order if all item sizes are close to 1/3. Interestingly, it gives a strong lower bound on the absolute random order ratio as well.

Finally, we revisit the case of 1/3-large items. To prove Proposition 1.3, we need to construct a list I with 1/3-large items and $\mathrm{E}[\mathrm{BF}(I^\sigma)] > \frac{6}{5}\,\mathrm{OPT}(I)$. Due to space restrictions, we only sketch the construction here and will provide the entire analysis in the full version of this paper.

Proof sketch of Proposition 1.3. We construct a list of $k = 3$ LM-pairs. For sufficiently small $\varepsilon > 0$ and $i \in [k]$ define $l_i = \frac{1}{2} + i\varepsilon$ and $m_i = \frac{1}{2} - i\varepsilon$. This way, $l_1 < l_2 < l_3$ and $m_1 > m_2 > m_3$. Clearly, $\mathrm{OPT}(I) = 3$. We can show that Best Fit uses 4 instead of 3 bins in at least 440 permutations. Therefore,

$$\frac{\mathrm{E}[\mathrm{BF}(I^\sigma)]}{\mathrm{OPT}(I)} \geq \frac{\frac{1}{6!} \cdot (440 \cdot 4 + (6! - 440) \cdot 3)}{3} = \frac{65}{54} > \frac{6}{5}\,. \qquad \blacktriangleleft$$

References

1. A. Al-Herz and A. Pothen. A 2/3-approximation algorithm for vertex-weighted matching. *Discrete Applied Mathematics*, 2019. (in press).
2. J. Balogh, J. Békési, G. Dósa, L. Epstein, and A. Levin. A new and improved algorithm for online bin packing. In *Proceedings of the 26th Annual European Symposium on Algorithms (ESA)*, volume 112 of *LIPIcs*, pages 5:1–5:14, 2018.
3. J. Balogh, J. Békési, G. Dósa, L. Epstein, and A. Levin. A new lower bound for classic online bin packing. In *Approximation and Online Algorithms - 17th International Workshop, (WAOA)*, volume 11926 of *Lecture Notes in Computer Science*, pages 18–28. Springer, 2019.
4. J. Boyar, G. Dósa, and L. Epstein. On the absolute approximation ratio for first fit and related results. *Discret. Appl. Math.*, 160(13-14):1914–1923, 2012.
5. M. G. Christ, L. M. Favrholdt, and K. S. Larsen. Online bin covering: Expectations vs. guarantees. *Theor. Comput. Sci.*, 556:71–84, 2014.
6. H. I. Christensen, A. Khan, S. Pokutta, and P. Tetali. Approximation and online algorithms for multidimensional bin packing: A survey. *Comput. Sci. Rev.*, 24:63–79, 2017.
7. E. G. Coffman, J. Csirik, G. Galambos, S. Martello, and D. Vigo. Bin packing approximation algorithms: survey and classification. In *Handbook of combinatorial optimization*, pages 455–531. Springer New York, 2013.

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

9 G. Dósa and J. Sgall. First fit bin packing: A tight analysis. In *30th International Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 20 of *LIPIcs*, pages 538–549, 2013.

10 G. Dósa and J. Sgall. Optimal analysis of best fit bin packing. In *Proceedings of the 41st International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 429–441, 2014.

11 C. Fischer and H. Röglin. Probabilistic analysis of the dual next-fit algorithm for bin covering. In *LATIN 2016: Theoretical Informatics - 12th Latin American Symposium*, pages 469–482, 2016.

12 C. Fischer and H. Röglin. Probabilistic analysis of online (class-constrained) bin packing and bin covering. In *LATIN 2018: Theoretical Informatics - 13th Latin American Symposium*, volume 10807 of *Lecture Notes in Computer Science*, pages 461–474. Springer, 2018.

13 B. Gamlath, M. Kapralov, A. Maggiori, O. Svensson, and D. Wajc. Online matching with general arrivals. In *60th IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, pages 26–37, 2019.

14 M. R. Garey, R. L. Graham, and J. D. Ullman. Worst-case analysis of memory allocation algorithms. In *Proceedings of the 4th Annual ACM Symposium on Theory of Computing (STOC)*, pages 143–150, 1972.

15 M. R. Garey and D. S. Johnson. "Strong" NP-completeness results: Motivation, examples, and implications. *J. ACM*, 25(3):499–508, 1978.

16 R. L. Graham. Bounds on multiprocessing timing anomalies. *SIAM Journal of Applied Mathematics*, 17(2):416–429, 1969.

17 A. Gupta and S. Singla. Random-order models. *CoRR*, abs/2002.12159, 2020. `arXiv:2002.12159`.

18 R. Hoberg and T. Rothvoss. A logarithmic additive integrality gap for bin packing. In *Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2616–2625, 2017.

19 Z. Huang, N. Kang, Z. G. Tang, X. Wu, Y. Zhang, and X. Zhu. How to match when all vertices arrive online. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 17–29, 2018.

20 Z. Huang, B. Peng, Z. Gavin Tang, R. Tao, X. Wu, and Y. Zhang. Tight competitive ratios of classic matching algorithms in the fully online model. In *Proceedings of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2875–2886, 2019.

21 Z. Huang, Z. G. Tang, X. Wu, and Y. Zhang. Online vertex-weighted bipartite matching: Beating 1-1/e with random arrivals. *ACM Trans. Algorithms*, 15(3):38:1–38:15, 2019.

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

23 D. S. Johnson, A. J. Demers, J. D. Ullman, M. R. Garey, and R. L. Graham. Worst-case performance bounds for simple one-dimensional packing algorithms. *SIAM J. Comput.*, 3(4):299–325, 1974.

24 E. G. Coffman Jr., J. Csirik, L. Rónyai, and A. Zsbán. Random-order bin packing. *Discret. Appl. Math.*, 156(14):2810–2816, 2008.

25 E. G. Coffman Jr. and G. S. Lueker. *Probabilistic analysis of packing and partitioning algorithms*. Wiley-Interscience series in discrete mathematics and optimization. Wiley, 1991.

26 N. Karmarkar and R. M. Karp. An efficient approximation scheme for the one-dimensional bin-packing problem. In *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 312–320, 1982.

27 R. M. Karp, U. V. Vazirani, and V. V. Vazirani. An optimal algorithm for on-line bipartite matching. In Harriet Ortiz, editor, *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing (STOC)*, pages 352–358, 1990.

28 C. Kenyon. Best-fit bin-packing with random order. In *Proceedings of the 7th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 359–364, 1996.

29 C. C. Lee and D. T. Lee. A simple on-line bin-packing algorithm. *J. ACM*, 32(3):562–572, 1985.

30 D. A. Levin and Y. Peres. *Markov chains and mixing times*, volume 107. American Mathematical Soc., 2017.

31 M. Mahdian and Q. Yan. Online bipartite matching with random arrivals: an approach based on strongly factor-revealing LPs. In *Proceedings of the 43rd ACM Symposium on Theory of Computing (STOC)*, pages 597–606, 2011.

32 A. Mehta. Online matching and ad allocation. *Foundations and Trends in Theoretical Computer Science*, 8(4):265–368, 2013.

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

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

35 P. V. Ramanan, D. J. Brown, C. C. Lee, and D. T. Lee. On-line bin packing in linear time. *J. Algorithms*, 10(3):305–326, 1989.

36 S. S. Seiden. On the online bin packing problem. *J. ACM*, 49(5):640–671, 2002.