Single-Sample Prophet Inequalities for Knapsack Problems

Pranav Nuti[1], Rebecca Reiffenhäuser[2], and Alice Sayutina[2]

[1]University of Chicago, Booth School of Business
[2]University of Amsterdam

Abstract

Prophet inequalities are a central tool for designing online algorithms with guarantees close to those obtained offline by an omniscient *prophet*. Those results are fuelled by assuming that the decision-maker has access to distributional knowledge of the future input. Recently, however, this assumption has been successfully weakened for several online problems, obtaining similar approximations with access to just a constant number of samples from the input distribution.

We investigate single-sample prophet inequalities for knapsack problems. Here, the decision-maker chooses a set of online rewards, which must not exceed a certain total weight. For the fractional knapsack problem, we give a 2-competitive algorithm, matching the best-possible ratio even with full distributional knowledge.

For integral knapsack, we give a 4-competitive single-sample prophet inequality, close to the state of the art under full distributional knowledge. We avoid the large-supply assumption otherwise common in work on such selection problems with samples, establishing the first close-to-optimal guarantees for the central class of *packing problems* in the single-sample regime. To demonstrate the versatility and power of the approach, we also investigate other problem settings and variants.

1 Introduction

1.1 Motivation

We consider a stochastic online problem where items with *values* and *weights* arrive in an online fashion (in a potentially adversarial order), where some prior information about the items is known from the start. The algorithm or decision-maker must, for every arriving item, immediately decide whether to include it in its solution, or discard it forever. The algorithm's objective is to maximize the sum of values over items in the solution. However, it must adhere to a maximum total weight of items, i.e., a knapsack size which cannot be exceeded.

While in absence of prior information, it is easy to see that a nontrivial approximation to the best value in hindsight is impossible, the famous *prophet inequality* paradigm allows for small constant competitive ratios. In this model, introduced first in [KS77]), it is assumed that the decision-maker initially knows the *independent distributions* from which the weight and value of every online item will be drawn. This allows for the computation of acceptance thresholds for each item that will ensure (in expectation) that the decision-maker obtains a constant fraction of the maximum reward possible if they knew all realizations of values and weights beforehand (i.e., for an omniscient *prophet*) [SC84].

While powerful in terms of enabling small competitive ratios, a serious downside of such prophet inequalities lies in the requirement of full distributional knowledge, which can rarely be fulfilled in practice. Bridging between this classic prophet inequality setting (practically unrealistic) and the setting without priors (no non-trivial guarantees) has fuelled a new line of research considering *much more practical algorithms*, requiring only *minimal* priors. This leads to our main question:

What competitive guarantees can be achieved for online knapsack problems when only a *minimum* amount of prior information, i.e., a single sample from each distribution, is known?

Arguably, samples are the most natural way in which distributional information can occur in practical settings (usually taking the form of collected, historical data). However, classic prophet inequalities in general cannot be learned from samples due to the possibility of rare, high-value events. Circumventing both this impossibility and the need to apply machine learning paradigms altogether, single-sample prophet inequalities (SSPIs) were first introduced by [AKW14]. They have since been studied for a range of online problems with constraints, including matchings, many matroid constraints ([CDF$^+$22, KNR22]), XOS combinatorial allocation [DKL$^+$24], and packing-type constraints under a large capacity/small weights assumption [GSW25]. However, for knapsack (and related, packing-type) problems without a small weights assumption, the development of SSPIs has lagged behind. Only very recently, an algorithm due to [Par25] for the classic knapsack problem establishes that in principle, achieving constant competitiveness with SSPI is possible (their algorithm is 64-competitive).

We present the first results for such settings with small constant competitive ratios, not far removed from those achievable when assuming full knowledge of the distributions, investigating various problem versions and settings.

1.2 Our results and techniques

Our first main result shows existence of a simple, optimal 2-competitive SSPI for the fractional knapsack problem. Our algorithm, which uses only a single sample, is optimal *even* amongst algorithms with full knowledge of the distributions.

Algorithm 1 A 2-competitive SSPI for fractional knapsack

1: **Input:** Sample items with weights and values, and capacity W.
2: Compute the fractional knapsack optimum on the samples.
3: Place the selected sample weight on the interval $[0, W]$, filling unused capacity with density 0 items.
4: Draw $U \sim \text{Unif}[0, W]$, and let ρ^* be the density at location U.
5: When the items arrive in an online fashion, accept the maximum feasible fraction of each arriving item which has a density larger than ρ^*.

Theorem 1. *Algorithm 1 is a 2-competitive SSPI for the fractional knapsack problem. No algorithm guarantees a competitive ratio better than* 2.

The impossibility of beating 2 is well-known (even with full knowledge of the distributions), and follows from a simple example with two items. Our result matches the competitive ratio previously known only under full knowledge of the distributions [DFKL20], and vastly improves upon the 64-competitive ratio of [Par25].

A major challenge in establishing this optimal (and not merely some constant) result is that unlike the algorithms and analyses in the small weights regime [KTRV14, GSW25], we cannot rely on concentration phenomena to argue that if we make good decisions on average using the sampled data, then we necessarily obtain a near-optimal competitive ratio.

Our algorithm instead generalizes the algorithm of [RWW19] for the single-choice SSPI problem (where all the items have weight 1, and the capacity of the knapsack is also 1), which sets the value of the highest sample as a threshold, and accepts the first online item that has value more than that threshold. Our central contribution is discovering the right way to generalize the SSPI algorithm for the single-choice problem: other natural generalizations do not seem to work.

The algorithm starts by computing the fractional optimum on the sampled items, and fills an imaginary knapsack with these items. It then selects a random item in this imaginary knapsack with

probability proportional to the weight the item occupies. Finally, it uses the density of that item as a threshold, accepting all items that arrive online with larger density.

Once we normalize the capacity of the knapsack to 1, the above procedure ensures that the probability for the threshold it sets to exceed any particular density level x is exactly equal to the expected amount of weight the prophet accepts with density greater than x. This exact same property is also satisfied by the algorithm for single-choice SSPI by [RWW19].

Our analysis of this algorithm is structurally similar to Ezra's analysis of the single-choice algorithm [Ezr26]. It proceeds via a *stochastic dominance* argument. We show for any x, that the items our algorithm accepts with density more than x have an expected weight of at least half the weight of the items the prophet accepts with density more than x.

The analysis then splits into two cases, according to whether the sampled threshold s lies above or below x. The high-threshold case is straightforward, because all weight the algorithm accepts has density at least s and hence at least x.

The main new idea in the analysis involves the low-threshold case, when s lies *below* x, and we would like to estimate the weight of the items the algorithm accepts with density *above* x. The danger is that our algorithm may spend its capacity on medium-density items with densities in $(s, x]$, and these items can crowd out the high-density items that matter for level x. We handle this by applying the FKG inequality [FGK71] with a carefully chosen ordering over each item's outcomes to show that the residual capacity after accepting the high-density items is negatively correlated with residual capacity from medium-density items. Combining the high- and low-threshold cases gives us a competitive ratio of 2. Our next result is for the integral version of the problem.

Theorem 2. *There is a 4-competitive SSPI for the integral knapsack problem.*

This result is a corollary of Theorem 1, obtained via a standard splitting argument to move from a fractional to an integral solution. There is only one *last* item that our algorithm in the fractional setting accepts partially. We obtain our guarantee by randomizing over two possible algorithms. We either imitate the fractional algorithm and fill the knapsack until we reach its last item, which we reject. Or else, we pack no items at all until the last item, and choose only the said last item. Note that this algorithm is not known to be optimal, and with full knowledge of distributions, an online contention resolution scheme based approach is known to obtain a competitive ratio of 3.135 [JMZ22].

We also give the following lower bound proving a clear gap in the optimal competitive ratio achievable in the fractional vs. integral knapsack problems:

Theorem 3. *For every $\varepsilon > 0$, there is an integral knapsack instance such that every online algorithm* ALG *(with or without distributional knowledge) satisfies:*

$$\mathbb{E}[\mathsf{OPT}] \geq \left(\frac{5}{2} - \varepsilon\right) \mathbb{E}[\mathsf{ALG}].$$

Our impossibility result in Theorem 3 is based on a two-stage instance that forces the algorithm to trade off a guaranteed value against access to more valuable future items. A collection of small, deterministic, identical "safe" items arrives first, followed by pairwise incompatible random "upgrade" items of increasing weight. The instance is set up so that if the algorithm accepts m safe items (with value m/k), the probability that any upgrade item still fits in the remaining capacity is $1-m/k$. Thus, every online algorithm obtains expected value at most 1, even with full knowledge of the distributions. The prophet, on the other hand, selects the lightest upgrade item and fills the remaining capacity with safe items, obtaining a value of $\frac{5}{2}$.

We also study some special cases where we can do better than for the integral knapsack problem. First, suppose we have some moderately large bound on the size of the individual items:

Theorem 4. *If every item's weight is bounded by cW, there is a $\frac{2}{1-c}$-competitive SSPI for the integral knapsack problem.*

To establish this result, we employ the simple observation that if each items weight is at most cW, we can fill a $1-c$ fraction of our knapsack using our fractional algorithm (run with weight $(1-c)W$) *and* still manage to take the last item the fractional algorithm accepts. Next, we have:

Theorem 5. *If every item's weight is either* 1 *or* 2*, and the capacity of the knapsack is* 2*, there is a* 3*-competitive SSPI.*

The 1-2 knapsack problem arises as a special case in [ALSS22] who study the knapsack problem in the *secretary* setting, and it is the first non-trivial generalization of the k-choice prophet inequality (generalizing both the single-choice prophet inequality problem (if all $w_i = 2$), and the two-choice prophet inequality problem (if all $w_i = 1$)). Our result illustrates that when the structure of possible item sizes is simple, one can achieve competitive ratios lower than 4 without resorting to a restriction of the largest item's size. The algorithm randomizes between the optimal algorithms for the single-choice and two-choice prophet inequality problems [RWW19, PS25].

Finally, we study the multiple knapsack setting where there are k identical knapsacks, and each item must be placed in one of the knapsacks. We obtain:

Theorem 6. *There is a* $min(4, \alpha(k))$*-competitive algorithm for the* k*-multiple knapsack problem, where* $\alpha(k) = \frac{2\lceil 1.7k+1.6 \rceil}{k}$.

For medium values of k, this provides an improvement over 4 without relying on a large capacity assumption. One way to understand our proof of Theorem 2 is that we started with a fractional solution to the problem, packed all the items even partially accepted by our fractional algorithm into two virtual knapsacks, and then chose one of the two at random to actually accept. Our algorithm for the multiple knapsack problem is a more sophisticated version of this idea: we start with a fractional solution to the problem, pack all the items even partially accepted by our fractional algorithm into $f(k) < 2k$ virtual knapsacks, and then choose k of them uniformly at random to actually accept. To pack all the items fractionally accepted into $f(k)$ knapsacks, we use the First-Fit algorithm from the bin-packing literature.

Note that another advantage of our algorithms is that we only need to know the set of samples to make our decisions, and not the identity of the distribution from which each sample is drawn. Furthermore, we allow an adversary to present the online realizations in an arbitrary order, chosen even after observing the samples.

Summarizing, we derive optimal and near-optimal SSPIs for knapsack and packing-type problems, a class previously neglected in this literature. Our results show that the structure of such settings allows for small constant ratios, even matching the full distributional knowledge setting for the fractional knapsack problem, while overcoming the issues caused by large items without resorting to large-capacity assumptions. See also our conclusions in section 8.

1.3 Further related work

Prophet inequalities were first considered by [Dyn63] and [KS77], and the optimal competitive ratio of 2 for the single-choice problem is accordingly long-known. Since the first 2-competitive threshold algorithm was derived by [SC84], prophet inequalities have developed into a rich area, with many problem variants investigated. Here, we only consider the directions closest to this paper. For more background, we suggest the surveys [Luc17], [CFH$^+$19], [HK92].

Prophet Inequalities One of the most central generalizations to the classic single-choice problem, and closely related to our settings, are matroid constraints. In [KW12], an optimal 2-competitive prophet inequality was introduced for the setting where the algorithm can choose any independent set of a given matroid. Their algorithm proceeds via recomputing acceptance thresholds after every step based on conditional expectations.

Besides considering different constraints and settings, major efforts have also gone into the development of simpler classes of mechanisms. [CHMS10], [CGKM20], [PS26] consider non-adaptive prophet inequalities, where prices for each arriving element are posted in advance and the item is accepted if it meets the threshold and it is still feasible to accept; or constrained non-adaptive mechanisms where the element must additionally meet an algorithm-determined sub-constraint.

A central concept in prophet inequalities are online contention resolution schemes (OCRS), introduced by [FSZ21]. Their OCRS imply constrained non-adaptive prophet inequality mechanisms, specifically obtaining an 11.65-competitive OCRS mechanism for a knapsack constraint. This competitive ratio was later improved to 3.14 via a new OCRS by [JMZ22], which is tight within the OCRS class of mechanisms and to the best of our knowledge, obtains the current state-of-the-art prophet inequality for the knapsack problem.

For the special case that no single item is larger than half the knapsack, [DFKL20] gave an improved $(3+\epsilon)$-competitive mechanism (with their approach yielding also a $5+\epsilon$ for the general problem). As a corollary, they derive a tight 2-approximation for the fractional knapsack problem.

Single-Sample Prophet Inequalities All the results above assume full knowledge of the distributions. Recently, there is growing success in massively weakening this assumption to make the decision-maker's algorithm more practical. A common model is that instead of the whole distribution, the decision-maker has access to one (or a small number of) sample(s) from each item's distribution. The first single sample prophet inequalities (SSPI) were introduced in [AKW14] for certain matroids and constant degree bipartite matching. Constant ratios were subsequently also derived for various matching problems and XOS combinatorial assignment, see [CDF$^+$22, KNR22, DKL$^+$24]. A large-supply variant, as is known to circumvent problems arising from large items in various settings for integral packing problems, was recently considered by [GSW25]. They achieve $1-\varepsilon$-competitive ratio for a general set of problems reducible to online resource allocation with large budgets.

Most relevant to our work, for the classic single-choice problem of choosing the largest online element, a celebrated result of [RWW19] showed that the optimal SSPI achieves the same competitive ratio of 2 as the best full-information prophet inequality. An alternative proof was given by [Ezr26]. The same ratio was later extended to also hold for the 2-item version (i.e. the algorithm can choose any set of at most two items) by [PS25], based on the same algorithmic idea. Recently, and solving a conjecture by [PS25], [NW26] proved a 2-competitive SSPI for any k-item prophet inequality.

For knapsack problems, to the best of our knowledge no SSPI results exist except for a new work of [Par25], who establish that constant ratios can in principle be achieved, by giving a 64-competitive single-sample prophet inequality under a knapsack constraint.

Other Relevant Work Online selection subject to a knapsack constraint was investigated also in the secretary, or random-order setting. Here, the distribution of the items' values and weights are unknown, but the items will arrive in uniformly random order. This setting has strong ties to SSPI for the special case that algorithms do not exploit the randomness of arrival after a so-called *sampling phase*, see [AKW14, CDF$^+$22]. Here, [BIKK07] gave a $10e \approx 27.18$-competitive secretary algorithm, improved subsequently by [KTRV14] to 8.06. The work by [AKL21] obtains a 6.65-competitive mechanism, further improved by [ALSS22] to e-competitive for special case of 1-2-knapsack. Finally, [KK25] improve the general secretary knapsack to 6.52-competitive, while also obtaining an e-competitive mechanism for the fractional variant.

In non-online settings, the multiple knapsack problem was studied in [Kel99], [CK05], and a FPTAS was introduced. Prophet inequalities for multiple knapsack were investigated in [JMZ22], obtaining the same guarantee of 3.14 as for single knapsack via an LP rounding technique.

Note that the problem of prophet inequalities with multiple knapsack constraints is closely connected with the bin packing problem. The state of the art guarantees for online bin packing are achieved by algorithms of the Harmonic family, originally introduced by [LL85], with the most currently advanced algorithm by [BBD$^+$17]. In our results, we make use of the *First Fit* algorithm class,

specifically the work of [GGJY76]. The original first fit analysis was given by [Ull71], and the tight analysis can be found in [DS13].

Several other notions of stochastic knapsack problems were considered. Packing integer programs, which generalize knapsack constraint to multiple linear constraints, in conjunction with size-stochasticity and adaptivity gaps were investigated in [DGV05]. A size-stochastic knapsack combined with OCRS techniques was considered in [YK23]. For other variations of online knapsack problem, we point the reader to the survey of [BHK+26].

2 Preliminaries

In the SSPI knapsack problem, there are n adversarially-chosen item-distributions $\mathcal{D}_1, \dots \mathcal{D}_n$. Each $\mathcal{D}_i$ is a distribution over value-weight pairs in $\mathbb{R}_{\geq 0} \times \mathbb{R}_{>0}$. The distributions are independent across items, but we do not assume that the value and weight of a single item are independent of each other. For an observation q_i from $\mathcal{D}_i$, we let $v(q_i)$, $w(q_i)$, and $\rho(q_i) = v(q_i)/w(q_i)$ denote its value, weight, and density respectively. By breaking ties with a tiny random perturbation if necessary, we will always assume that the distribution $\mathcal{D}_i$ are continuous.

In the single-sample model, the algorithm does not have knowledge of the distributions $\mathcal{D}_i$. Instead, it has access to a single sample $s_i \sim \mathcal{D}_i$ from each item distribution. Note that we only assume that the algorithm knows the set of samples s_i (and does not need to know the identity of the distribution from which each sample is drawn).

In the online phase of the problem, realizations $r_i \sim \mathcal{D}_i$ are drawn from the distributions. We assume that the pairs s_i and r_i are independent of each other. Before the online realizations are revealed, an adversary may choose an arbitrary permutation π. The algorithm then observes $r_{\pi(1)}, \dots, r_{\pi(n)}$, one-by-one in an online fashion, and must irrevocably decide, upon each arrival, whether (and potentially, how much) of that item to accept.

The *knapsack* has *capacity* denoted by W, and we typically normalize this capacity, and set $W = 1$. An algorithm ALG for the SSPI *fractional* (resp. *integral*) knapsack problem assigns to each realization r_i (in an online, irrevocable manner) a fraction $\alpha_i \in [0, 1]$ (resp. $\{0, 1\}$), and the assignment is feasible if $\sum_i \alpha_i w(r_i) \leq W$, in which case its value is $\mathsf{ALG} = \sum_i \alpha_i v(r_i)$. A valid algorithm always makes a feasible assignment.

The expected performance of an algorithm is $\mathbb{E}_{\vec{r},\vec{s}}[\mathsf{ALG}]$, computed by taking an expectation over both $\vec{s}$ and $\vec{r}$, and any internal randomness the algorithm uses. We benchmark the expected performance of an algorithm by comparing it to the expected value obtained by a so-called prophet, an omniscient entity who observes all of the r_i at once, and chooses the best possible feasible assignment in hindsight. We denote by FOPT (resp. OPT) the value of this offline optimum in the fractional (resp. integral) setting. For $\alpha \geq 1$, a fractional knapsack SSPI is α-competitive if, for every choice of item-distributions,

$$\mathbb{E}[\mathsf{ALG}] \geq \frac{1}{\alpha}\mathbb{E}[\mathsf{FOPT}],$$

For the integral version of the problem, the benchmark is OPT in place of FOPT. In the integral version of the problem we will assume that the weights of all the items are at most W, since items with weight more than W cannot be accepted by OPT either.

Finally, we consider a variant with multiple knapsacks. In the integral multiple-knapsack problem, the decision-maker has k identical knapsacks, each of capacity W. Upon observing a realization r_i, the algorithm must irrevocably reject it or assign it to one of the k knapsacks. An assignment is feasible if every item is assigned to at most one knapsack and, for each $j \in [k]$, the total weight assigned to knapsack j is at most W. Its value is the total value of all assigned items. The samples, online arrival model, and prophet benchmark are as defined above, and we continue to denote the integral offline optimum by OPT.

3 A 2-competitive SSPI for fractional knapsack

In this section, we consider the fractional version of the SSPI knapsack problem where when an item arrives, the decision-maker may accept any fraction of the item. We analyze algorithm 2:

Algorithm 2 A 2-competitive SSPI for fractional knapsack

1: **Input:** Sample items $s_1, s_2, \ldots, s_n$, total allowed weight W.
2: Compute the fractional knapsack optimum on the samples.
3: Place the selected sample weight on the interval $[0, W]$, filling unused capacity with density 0 items.
4: Draw $U \sim \text{Unif}[0, W]$, and let ρ^* be the density at location U.
5: When the items arrive in an online fashion, accept the maximum feasible fraction of each arriving r_i with $\rho(r_i) > \rho^*$.

Our analysis is structurally similar to Ezra's analysis of [Ezr26], although we have to take into account the possibility of selecting multiple items. A key element to our solution is to apply a correlation inequality to estimate the chance that picking some medium-value items stops us from picking high-value items.

Theorem 1. *Algorithm 2 is a 2-competitive SSPI for the fractional knapsack problem.*

Proof. We normalize the capacity of the knapsack to 1. Formally speaking, we break ties between the added 0 density items by adding a tiny random perturbation. Our plan is to apply the well known 'stochastic-dominance' strategy for establishing prophet inequalities.

The stochastic dominance idea. For a density threshold $x \geqslant 0$, define:

$$P_x = \sum_i w(r_i)\mathbf{1}\{\rho(r_i) > x\}.$$

Thus, the prophet's total selected weight with density larger than x is $\min\{1, P_x\}$. Let

$$q(x) = \mathbb{E}\left[\min\{1, P_x\}\right].$$

By the 'layer cake representation' for the value obtained by the prophet, letting α_i denote the fraction of item i the prophet accepts, we get:

$$\begin{aligned}
\mathbb{E}[\mathsf{FOPT}] &= \mathbb{E}\left[\sum_i \alpha_i w(r_i)\rho(r_i)\right] \\
&= \mathbb{E}\left[\int_0^\infty \sum_i \alpha_i w(r_i)\mathbf{1}\{\rho(r_i) > x\}dx\right] \\
&= \int_0^\infty \mathbb{E}\left[\sum_i \alpha_i w(r_i)\mathbf{1}\{\rho(r_i) > x\}\right] dx \\
&= \int_0^\infty q(x)dx.
\end{aligned}$$

Let A_x be the total weight our algorithm accepts whose density is larger than x. Again by the layer cake representation,

$$\mathbb{E}[\mathsf{ALG}] = \int_0^\infty \mathbb{E}[A_x]dx.$$

Thus, we see that it suffices to show that for every $x > 0$,

$$\mathbb{E}[A_x] \geq \frac{1}{2}q(x). \tag{1}$$

Our algorithm's threshold. The main idea our algorithm exploits is that the sample threshold is chosen exactly so that:

$$\Pr[\rho^* > x] = q(x).$$

Indeed, the probability that the random location U falls on a sample weight of density larger than x is exactly the amount of weight of density larger than x in the sampled fractional optimum. Since samples and realizations have the same distribution, the claimed equality follows. Note that in the single-choice SSPI, the usual threshold of the maximum of the samples has exactly the same property.

Now fix an x for which we wish to establish (1), and let

$$F(x) := \Pr[\rho^* \leqslant x] = 1 - q(x) = 1 - \mathbb{E}\left[\min\{1, P_x\}\right] = \mathbb{E}[(1 - P_x)^+],$$

be the CDF of the threshold. Imitating the single-choice setting, we condition on the algorithm's threshold ρ^* being equal to s and split the analysis according to whether s lies above or below x.

The high threshold case. First suppose $s \geqslant x$. Then every item accepted by the algorithm has density larger than x, so the algorithm accepts exactly $\min\{1, P_s\}$ total weight of density larger than s. Therefore

$$\mathbb{E}[A_x \mid \rho^* = s] = q(s) = 1 - F(s).$$

Noting that the CDF of the threshold is also $F(x)$, and integrating over the possible thresholds $s \geqslant x$ gives:

$$\mathbb{E}\left[A_x \mathbf{1}\{\rho^* \geqslant x\}\right] = \int_x^\infty (1 - F(s)) dF(s) = \frac{(1 - F(x))^2}{2}.$$

This result is analogous to the $s \geq x$ case in the single-choice setting.

The low threshold case. Now suppose the threshold the algorithm sets is $s < x$. Our goal will be to show that:

$$\mathbb{E}[A_x \mid \rho^* = s] \geq \frac{F(s)}{F(x)} - F(s), \tag{2}$$

matching the bound from the single-choice setting. Let

$$H = P_x = \sum_i w(r_i) \mathbf{1}\{\rho(r_i) > x\}, \quad M = \sum_i w(r_i) \mathbf{1}\{s < \rho(r_i) \leqslant x\}.$$

Thus H is the high-density weight and M is the medium-density weight, with respect to the two thresholds s and x. The principal danger our algorithm faces due to setting a threshold of s is that it accepts too much of the medium density items: these are the only items that potentially stop the algorithm from taking the high-density items. Therefore, we conclude that,

$$A_x \geqslant \min\{H, (1 - M)^+\} = (1 - M)^+ - (1 - H - M)^+. \tag{3}$$

Define

$$R_H = (1 - H)^+, \quad R_M = (1 - M)^+, \quad R_{H+M} = (1 - H - M)^+,$$

and note that $F(x) = 1 - q(x) = \mathbb{E}[R_H]$ and $F(s) = \mathbb{E}[R_{H+M}]$. Taking expectations in (3), we conclude that:

$$\mathbb{E}[A_x \mid \rho^* = s] \geq \mathbb{E}[R_M] - F(s).$$

Thus, in view of our goal (2), it remains to show that:

$$\mathbb{E}[R_M] \geq \frac{F(s)}{F(x)} = \frac{\mathbb{E}[R_{H+M}]}{\mathbb{E}[R_H]}.$$

Now pointwise, we know that:

$$R_{H+M} \leq R_H R_M,$$

because either the right hand side is 0, in which case the inequality is trivial, or the right hand side is positive, in which case the inequality just says that $1-H-M \leq 1-H-M+HM$. Taking expectations, we see that $\mathbb{E}[R_{H+M}] \leq \mathbb{E}[R_H R_M]$, so it remains to show that R_H and R_M are negatively-correlated.

Intuitively, this is clear because having more high-density items clearly hurts the medium-density items. To establish this formally, we use the FKG inequality. For each item i, order the possible outcomes as follows: outcomes with $\rho(r_i) > x$ come first, ordered by decreasing $w(r_i)$, then outcomes with $\rho(r_i) \leq s$; finally outcomes with $s < \rho(r_i) \leq x$, ordered by increasing $w(r_i)$. With respect to this order, R_H is increasing, while R_M is decreasing, so we may apply the FKG inequality to these two quantities to conclude:

$$\mathbb{E}[R_H R_M] \leqslant \mathbb{E}[R_H]\mathbb{E}[R_M].$$

Thus, we have achieved our goal and established that:

$$\mathbb{E}[A_x \mid \rho^* = s] \geq \frac{F(s)}{F(x)} - F(s) = \frac{((1-F(x))F(s)}{F(x)}.$$

Now integrating over the possible thresholds $s < x$ gives:

$$\mathbb{E}\left[A_x \mathbf{1}\{\rho^* < x\}\right] = \frac{1-F(x)}{F(x)} \int_0^x F(s) dF(s) = \frac{(1-F(x))F(x)}{2}.$$

This is analogous to the $s < x$ case in the single-choice setting.

Putting it all together. We conclude that:

$$\begin{aligned}
\mathbb{E}[A_x] &= \mathbb{E}\left[A_x \mathbf{1}\{\rho^* < x\}\right] + \mathbb{E}\left[A_x \mathbf{1}\{\rho^* \geq x\}\right] \\
&= \frac{(1-F(x))F(x)}{2} + \frac{(1-F(x))^2}{2} \\
&= \frac{q(x)}{2},
\end{aligned}$$

as desired. □

4 A 4-competitive SSPI for integral knapsack

We now use a standard reduction to turn the 2-competitive fractional algorithm into a 4-competitive algorithm for integral knapsack.

The 'last item' is just the first item whose virtual fractional acceptance makes the virtual knapsack full. If the virtual run never hits a capacity-exceeding item, then there is no last item: in the HEADS branch we accept all virtual items, and in the TAILS branch we accept nothing.

Theorem 2. *Algorithm 3 is 4-competitive SSPI for the integral knapsack problem.*

Proof. Let FALG be the value obtained by the virtual fractional run inside Algorithm 3. We first compare Algorithm 3 to the virtual fractional run. The virtual fractional run accepts some items fully, and possibly one last item r fractionally. Let R be the set of fully accepted items before this last item. Write $\alpha \in [0, 1)$ for the fraction of r accepted by the virtual run. Then

$$\mathsf{FALG} = v(R) + \alpha \cdot v(r).$$

The HEADS branch of our algorithm obtains $v(R)$, while the TAILS branch obtains $v(r)$. Therefore the expected value of our algorithm is:

$$\frac{1}{2}\mathbb{E}\left[v(R) + v(r)\right] \geq \frac{1}{2}\mathbb{E}\left[(v(R) + \alpha \cdot v(r))\right] = \frac{1}{2}\mathbb{E}[\mathsf{FALG}].$$

Algorithm 3 A 4-competitive algorithm for integral knapsack SSPI

Input: Sample items $s_1, s_2, \ldots, s_n$, total allowed weight W.
Compute the same sample threshold ρ^* as in Algorithm 2.
Toss a fair coin.
if HEADS **then**
    Run Algorithm 2 virtually on the online items.
    Accept every item that the virtual fractional run accepts fully before its last item.
    If the virtual run reaches a capacity-exceeding last item that it accepts only fractionally, reject that item and stop.
else
    Run Algorithm 2 virtually on the online items, accepting nothing before the virtual fractional run's last item.
    If the virtual run reaches a capacity-exceeding last item that it accepts only fractionally, accept that item and stop.
end if

Using the 2-competitive guarantee for the fractional algorithm,

$$\mathbb{E}[\mathsf{ALG}] \geq \frac{1}{2}\mathbb{E}[\mathsf{FALG}] \geq \frac{1}{4}\mathbb{E}[\mathsf{FOPT}] \geq \frac{1}{4}\mathbb{E}[\mathsf{OPT}].$$

Hence Algorithm 3 is 4-competitive. □

5 Improved algorithms for variants

5.1 A $\frac{2}{1-c}$-competitive SSPI for c-bounded integral knapsack

We now give a better-than-four competitive algorithm for integral knapsack, assuming each item is smaller than half the knapsack (i.e., $w(a_i) < 1/2$ for all a_i).

For this, recall that Algorithm 2 guarantees a competitive ratio of 2 for the fractional knapsack problem (see Theorem 1). Recall further that our algorithm for the integral case simply stops running Algorithm 2 once it attempts to add an item which no longer fully fits, or alternatively *only* seeks to add this last item, losing an additional factor of 2.

Observe that when the weight of each item is limited to be $\leq c$ for fixed $c \leq 1/2$, one can safely combine both: first, we can fill at most $(1-c)$ weight into the knapsack, and then also select one additional item without exceeding the total capacity of $W = 1$. We use this observation to design the following algorithm:

Algorithm 4 A $\frac{2}{1-c}$-competitive SSPI for c-bounded integral knapsack

Input: Sample items $s_1, s_2, \ldots, s_n$, capacity $W = 1$, and bound $c < 1$.
Run Algorithm 2 with $W = 1 - c$, accepting every item it accepts fully. If the run reaches a capacity-exceeding last item that it accepts only fractionally, accept that item and stop.

Theorem 4. *If no item exceeds weight $c < 1$, Algorithm 4 is a $\frac{2}{1-c}$-competitive SSPI for the integral knapsack problem.*

Proof. First, we note that the procedure never exceeds the knapsack capacity and therefore produces a feasible solution: while running Algorithm 2 with $W = 1 - c$, we use at most $(1-c)$ space until we reach the last item that Algorithm 2 accept, while the last item only adds a single item (i.e., weight $\leq c$) at most.

Let $\mathsf{FALG}(1-c)$ be the value obtained by the run of Algorithm 2. Let R be the set of items it fully accepts, before its last item r. Write $\alpha \in [0,1)$ for the fraction of r accepted by the virtual run. Then

$$\mathsf{ALG} = v(R) + v(r) \geq v(R) + \alpha v(r) = \mathsf{FALG}(1-c).$$

As shown in Theorem 1, $\mathsf{FALG}(1-c)$ is a 2-competitive algorithm with respect to the fractional offline optimum on a knapsack of capacity $(1-c)$. Therefore,

$$\mathbb{E}[\mathsf{FALG}(1-\mathsf{c})] \geq \frac{1}{2}\mathbb{E}[\mathsf{FOPT}(1-\mathsf{c}).]$$

Now note that $\mathbb{E}[\mathsf{FOPT}(1-\mathsf{c})] \geq (1-c)\mathbb{E}[\mathsf{FOPT}]$. This inequality holds because the fractional optimum with $1-c$ total weight is obtained by simply ordering items by decreasing density, and taking the first $(1-c)$ weight. Any fractional solution on a larger knapsack will only deviate at some point to take some lower density than the fractional optimum on a lower capacity knapsack.

We conclude that:

$$\mathbb{E}[\mathsf{ALG}] \geq \mathbb{E}[\mathsf{FALG}(1-\mathsf{c})] \geq \frac{1}{2}\mathbb{E}[\mathsf{FOPT}(1-\mathsf{c})] \geq \frac{1-c}{2}\mathbb{E}[\mathsf{OPT}].$$

Hence Algorithm 4 is $\frac{2}{1-c}$-competitive. □

As is easily seen, Theorem 4 offers competitive ratio better than 4 if no single item fills up half the knapsack. For example, when the largest item has weight 1/3, this yields competitive ratio 3, which for even smaller items quickly moves towards the ratio of 2 obtained by our Theorem 1 for the fractional case.

5.2 A 3-competitive SSPI for 1-2 integral knapsack

Consider now a 1-2 knapsack problem. The 1-2 knapsack is a special case of the knapsack SSPI, with the additional restriction that each item's weight is either 1 or 2, and the total allowed weight is $W = 2$. To design our algorithm, we use the algorithms from [RWW19] for $k = 1$, and [PS25] for $k = 2$ item selections, which both guarantee a competitive ratio of 2.

Algorithm 5 A 3-competitive SSPI for 1-2 integral knapsack

Input: Sample items $s_1, s_2, \ldots, s_n.$, total allowed weight $W = 2$
Toss a coin with $\Pr[\mathsf{HEADS}] = 2/3$, $\Pr[\mathsf{TAILS}] = 1/3$
if HEADS **then**
    Run a single-choice SSPI algorithm ignoring weights, i.e.,:
    $T \leftarrow \max_{i\in[n]} v(s_i)$
    Accept the first i with $v(r_i) \geqslant T$ and stop.
else
    Discard heavy items with $w_i = 2$, and run a 2-choice SSPI algorithm, i.e.,:
    $T \leftarrow \text{secondmax}_{i\in[n], w(s_i)=1} v(s_i)$
    Accept at most two items s.t. $w(r_i) = 1$, $v(r_i) \geqslant T$.
end if

Theorem 5. *Algorithm 5 is a 3-competitive SSPI for the 1-2 knapsack problem.*

Proof. Let E_1 be the event that OPT consists of one item, and E_2 the event that OPT consists of two items. Observe that:

$$\mathbb{E}[\mathsf{OPT}] = \mathbb{E}[v_j \mathbf{1}\{E_1\}] + \mathbb{E}[(v_{i_1} + v_{i_2})\mathbf{1}\{E_2\}],$$

where v_j is the largest realized value, v_{i_1} is the largest realized value with weight 1, and v_{i_2} is the second largest realized with weight 1 (by abuse of notation we let $v_{i_1} = 0$ or $v_{i_2} = 0$ if no such values exist).

Now observe $\mathbb{E}[\mathsf{ALG} \mid \mathsf{HEADS}] \geqslant \frac{1}{2}\mathbb{E}[v_j]$, because we use a two-competitive one-item SSPI. Then,

$$\mathbb{E}[\mathsf{ALG} \mid \mathsf{HEADS}] \geqslant \frac{1}{2}\mathbb{E}[v_j] \geq \frac{1}{2}\mathbb{E}[v_j \mathbf{1}\{E_1\}] + \frac{1}{2}\mathbb{E}[v_{i_1}\mathbf{1}\{E_2\}].$$

Similarly,

$$\mathbb{E}[\mathsf{ALG} \mid \mathsf{TAILS}] \geqslant \frac{1}{2}\mathbb{E}[v_{i_1} + v_{i_2}] \geqslant \frac{1}{2}\mathbb{E}[(v_{i_1} + v_{i_2})\mathbf{1}\{E_2\}].$$

Summing these expressions together:

$$\begin{aligned}
\mathbb{E}[\mathsf{ALG}] &= \frac{2}{3}\mathbb{E}[\mathsf{ALG} \mid \mathsf{HEADS}] + \frac{1}{3}\mathbb{E}[\mathsf{ALG} \mid \mathsf{TAILS}] \\
&\geq \frac{1}{3}\mathbb{E}[v_j \mathbf{1}\{E_1\}] + \frac{1}{3}\mathbb{E}[v_{i_1}\mathbf{1}\{E_2\}] + \frac{1}{6}\mathbb{E}[(v_{i_1} + v_{i_2})\mathbf{1}\{E_2\}] \\
&\geq \frac{1}{3}\mathbb{E}[v_j \mathbf{1}\{E_1\}] + \frac{1}{3}\mathbb{E}[(v_{i_1} + v_{i_2})\mathbf{1}\{E_2\}] \\
&= \frac{1}{3}\mathbb{E}[\mathsf{OPT}]
\end{aligned}$$

as desired. □

6 A 5/2-lower bound for integral knapsack

In this section, we provide a simple impossibility result for the integral knapsack prophet inequality problem. This impossibility result also holds when the online algorithm knows all the items' distributions. Throughout this section, we fix the capacity of the knapsack at $W = 1$.

Theorem 3. *For every $\varepsilon > 0$, there is an integral knapsack instance such that every online algorithm* ALG *(potentially with distributional knowledge) satisfies:*

$$\mathbb{E}[\mathsf{OPT}] \geq \left(\frac{5}{2} - \varepsilon\right)\mathbb{E}[\mathsf{ALG}].$$

Proof. Fix a (large) integer k. The hard instance has $2k$ items. First, there are k deterministic 'safe' items $a_1, \ldots, a_k$, each with

$$w(a_i) = \frac{1}{2k}, \quad v(a_i) = \frac{1}{k}.$$

Thus, all safe items together have total weight $1/2$ and total value 1. After the safe items, there are k random 'upgrade' items $b_1, \ldots, b_k$. For $j = 1, \ldots, k-1$,

$$w(b_j) = \frac{1}{2} + \frac{j}{2k}, \quad v(b_j) = \begin{cases} 1, & \text{with probability } \frac{1}{k-j+1}. \\ 0, & \text{otherwise.} \end{cases}$$

Finally, the last item has

$$w(b_k) = 1, \quad v(b_k) = \begin{cases} k, & \text{with probability } \frac{1}{k}. \\ 0, & \text{otherwise.} \end{cases}$$

We start by showing that every online algorithm has expected value at most 1. Suppose the online algorithm accepts m of the safe items, for some $m \in \{0, 1, \ldots, k\}$. If $m = 0$, the algorithm can accept

any of the upgrade items. If it chooses to accept before the last item, it gets value 1, and if it waits until the last item, it still gets value 1.

Now suppose instead that $m \geq 1$ and the algorithm accepts at least one of the safe items, accumulating a value of $\frac{m}{k}$. Then, the algorithm cannot accept the last item, and in fact, the algorithm can accept at most one of $b_1, b_2, \ldots, b_{k-m}$. Therefore, the value the algorithm can obtain from the upgrade items is at most the probability that one of the $b_1, \ldots, b_{k-m}$ is active:

$$\Pr[\text{one of } b_1, \ldots, b_{k-m} \text{ is active}] = 1 - \prod_{j=1}^{k-m} \left(1 - \frac{1}{k-j+1}\right) = 1 - \frac{m}{k}.$$

Thus, the total value the algorithm can obtain is once again at most 1.

Next, we calculate the expected value of the prophet. If b_k is active (with probability $\frac{1}{k}$), the prophet takes b_k alone and obtains a value of k. Since k is large, this is better than accepting any of the safe items or one of the other upgrade items.

Else conditioning on b_k being inactive, but one of the other upgrade items being active, the prophet accepts the active upgrade item with the smallest weight, and fills the rest of their knapsack with the safe items. The probability that b_i is the first upgrade item that is active is

$$\frac{1}{k-i+1} \prod_{j=1}^{i-1} \left(1 - \frac{1}{k-j+1}\right) = \frac{1}{k}.$$

Finally, if none of the upgrade items are active, the prophet accepts all the safe items and obtains value 1. We conclude that the expected value obtained by the prophet is:

$$\begin{aligned}
\mathbb{E}[\mathsf{OPT}] &= \frac{1}{k} \cdot k + \left(1 - \frac{1}{k}\right) \left(\sum_{i=1}^{k} \frac{1}{k} \left(1 + \frac{k-i}{k}\right)\right) \\
&= 1 + \left(1 - \frac{1}{k}\right) \left(2 - \sum_{i=1}^{k} \frac{i}{k^2}\right) \\
&= 1 + \left(1 - \frac{1}{k}\right) \left(2 - \frac{k+1}{2k}\right).
\end{aligned}$$

Taking $k \to \infty$ we see that $\mathbb{E}[\mathsf{OPT}]$ goes to $\frac{5}{2}$, as needed. □

7 Integral multiple knapsack

In this section, we consider the problem of integral multiple knapsack, as defined in the preliminaries., i.e., we assume that the decision-maker has at their disposal k identical knapsacks, each of total capacity 1. We show that if the number k of knapsacks is moderately large, we achieve an improvement in competitive ratio compared to the single knapsack setting.

Theorem 6. *There is a $min(4, \alpha(k))$-competitive algorithm for the k- multiple knapsack problem, where $\alpha(k) = \frac{2\lceil 1.7k+1.6 \rceil}{k}$.*

The rest of this section is dedicated to proving theorem 6. We start by noting that there is a 4-competitive SSPI for integral multiple knapsack, quite similar to our 4-competitive SSPI for integral knapsack. First, we pack a virtual knapsack of capacity k by running Algorithm 2, and place all (possibly fractionally) accepted items consecutively in the interval $[0, k]$, rounding up one item if necessary. Divide the interval into k unit-sized intervals. Then, each of the items in a single interval can be packed (integrally) into two knapsacks of capacity 1 (with items crossing intervals assigned to the knapsacks associated with the interval on their left).

This constructs, in a deterministic and online way, $2k$ virtual knapsacks of expected total value at least $\frac{1}{2}\mathbb{E}[\mathsf{OPT}]$. Before the online phase of the algorithm, choose a uniformly random k-sized subset of the $2k$ knapsacks, and use only the chosen knapsacks as actual knapsacks to be accepted. Each virtual knapsack is chosen with probability 1/2. Therefore:

$$\mathbb{E}[\mathsf{ALG}] \geq \frac{1}{2}\mathbb{E}[\mathsf{FALG}] \geq \frac{1}{4}\mathbb{E}[\mathsf{FOPT}] \geq \frac{1}{4}\mathbb{E}[\mathsf{OPT}].$$

Now, we present algorithm 6, which generalizes the procedure we have just discussed, using ideas from bin packing. The algorithm designer needs to specify three parameters: a function $w' \geq w$ that takes in an item's value and weight, and returns an adjusted weight, a restriction C on how much adjusted weight a fractional knapsack subroutine is going to accept, and $f(k)$, a function determining how many knapsacks are used for virtual (as opposed to actual) packing. The 4-competitive algorithm we discussed used $C = k$, $w' = w$, and $f(k) = 2k$.

The algorithm processes each item in two stages. First, it decides whether to accept the item into a virtual knapsack (in a 'fractional knapsack' stage) in which case we will say the item has been admitted, and second decides where to pack it (in a 'bin assignment' stage). Only a random subset of k of the virtual knapsack are actually accepted. We require that $f(k)$, w', and C are selected so that the 'bin assignment' stage of the algorithm always succeeds (this means that $f(k)$ is sufficiently large that the first-fit bin packing we apply uses at most $f(k)$ bins), and we will choose them later so that this is the case. For now, we will assume the stage succeeds.

Algorithm 6 A k-knapsack SSPI algorithm

Input: Sampled items $s_1, s_2, \ldots, s_n$, the number of knapsacks k.
Parameters: limit on admitted items' adjusted weight C, adjusted weight function w', $f(k)$ number of virtual knapsacks.
Run fractional knapsack algorithm on adjusted weights (rounding last item up) followed by bin assignment using First-Fit bin packing:
Let $B_1, B_2, \ldots, B_{f(k)}$ be virtual knapsacks of capacity 1.
$S \leftarrow$ random k-sized subset of $[f(k)]$ of actual knapsacks.
Define $\rho'(x) = v(x)/w'(x)$
$\rho^* \leftarrow$ threshold computed by fractional knapsack algorithm assuming values v, weights w', and capacity C.
$a \leftarrow 0$ ▷ Weight of all virtually accepted items.
for $i \in [n]$ **do**
    if $\rho'(r_i) > \rho^*$ and $a < C$ **then**
        $a \leftarrow a + w'(r_i)$ ▷ Item is admitted.
        $j \leftarrow$ smallest index such that $w(B_j) + w(r_i) \leqslant 1$.
        ▷ $f(k), w'$, and C chosen so that such a j exists.
        $B_j \leftarrow B_j \cup \{r_i\}$
        Accept item if $j \in S$
    end if
end for

Let I be the set of items admitted, or accepted into some virtual knapsack. For a fixed realization, let

$$\mathsf{IP} = \max\left\{\sum_{i=1}^{n} v(r_i)x_i : x \in \{0,1\}^n, \sum_{i=1}^{n} w'(r_i)x_i \leq C\right\},$$

be the integral offline optimum if items had adjusted weights w' and the capacity of the knapsack was C. We have the following lemma:

Lemma 1. $\mathbb{E}[\mathsf{ALG}] \geqslant \frac{\mathbb{E}[\mathsf{IP}]}{2(f(k)/k)}$.

Proof. Note first of all that the expected total value of admitted items is at least half the value of $\mathbb{E}[\mathsf{IP}]$, because the underlying fractional knapsack algorithm is 2-competitive. Then,

$$\mathbb{E}[\mathsf{ALG}] = \frac{k}{f(k)} \sum_{t=1}^{f(k)} \mathbb{E}[v(B_t)] = \frac{k}{f(k)} \mathbb{E}[v(I)] \geqslant \frac{\mathbb{E}[\mathsf{IP}]}{2f(k)/k}$$

□

Note that if W' is an upper bound on the w' function, then $w'(I) \leqslant C + W'$, since at most one item is accepted into a virtual knapsack beyond its capacity C.

7.1 A $3.5 + \frac{7.5}{k}$-competitive SSPI for multiple knapsack

To illustrate the main idea about how we control the number of virtual knapsacks used by First-Fit, we first present a simplified analysis which guarantees a competitive ratio of $\alpha(k) = 3.5 + 7.5/k$. The proof we present here follows a well-known, classic first-fit bin-packing analysis. We choose the following parameters:

$$C = \frac{7k}{6}, \quad W' = \frac{7}{6}, \quad f(k) = \left\lfloor \frac{15 + 7k}{4} \right\rfloor, \quad w'(x) = w(x) + \text{bonus}(x), \text{bonus}(x) = \frac{1}{6} \mathbf{1} \left\{ w(x) > \frac{1}{2} \right\}$$

First, we note that:

Lemma 2. $\mathsf{IP} \geqslant \mathsf{OPT}$.

Proof. Consider an optimal assignment OPT of items to knapsacks. Due to integrality, each knapsack contains at most one item of weight $> 1/2$. Thus, the adjusted weight w' in every knapsack is at most $\frac{7}{6}$. Thus, OPT is a feasible solution to the integer program defined above. □

Lemma 3. *With the exception of possibly two virtual knapsacks B_i, every non-empty B_i satisfies $w'(B_i) > \frac{2}{3}$.*

Proof. First, note that with the exception of possibly one virtual knapsack, every B_i satisfies $w(B_i) > \frac{1}{2}$. Indeed, if there were B_i and $B_{i'}$ such that $i < i'$, and $w(B_i) \leqslant \frac{1}{2}$, $w(B_{i'}) \leqslant \frac{1}{2}$, then we made an error while assigning items in $B_{i'}$, because we could put them in the earlier virtual knapsack B_i.

Now consider every other virtual knapsack B_j with $w(B_j) > \frac{1}{2}$. If B_j contains a single item, $w(B_j) > \frac{1}{2}$ implies that $w'(B_j) > \frac{2}{3}$. So suppose by way of contradiction that there are two virtual knapsacks with B_j and $B_{j'}$ such that $j < j'$, and $\frac{2}{3} \geq w(B_j) > \frac{1}{2}$, and each has at least two items.

Since $B_{j'}$ contains two items or more, at least one of those items is at most $\frac{w(B_{j'})}{2} \leqslant \frac{1}{3}$ weight. This item should have been assigned to B_j instead of $B_{j'}$. □

Lemma 4. *The first-fit bin assignment step of our algorithm always succeeds, and with the choice of parameters we defined, Algorithm 6 is $\alpha(k) := 3.5 + \frac{7.5}{k}$-competitive.*

Proof. The total weight w' of items accepted is at most $C + W' = 7/6 \cdot (k + 1)$. Thus, the number of virtual knapsacks actually used is at most $2 + \lfloor \frac{7/6 \cdot (k+1)}{2/3} \rfloor = \lfloor 15/4 + 7/4 \cdot k \rfloor = f(k)$, as needed. The competitive ratio follows immediately from lemma 1, and the value of $f(k)$. □

7.2 An improved SSPI for multiple knapsack

In this section, we present a choice of parameters for which Algorithm 6 obtains a competitive ratio of $\alpha(k) = \frac{2\lceil 1.7k + 1.6 \rceil}{k}$. This analysis follows the same design principle as the previous section, but the details are more complex. We use the bin packing strategy from [GGJY76]. Let:

$$C = 1.7k, \quad W' = 1.6, \quad f(k) = \lceil 1.7k + 1.6 \rceil, \quad w'(x) = 1.2w(x) + \text{bonus}(x),$$

where

$$\text{bonus}(x) = \begin{cases} 0 & \text{if } w(x) \leqslant 1/6 \\ \frac{3}{5}(w(x) - \frac{1}{6}) & \text{if } w(x) \in (\frac{1}{6}, \frac{1}{3}) \\ 0.1 & \text{if } w(x) \in [\frac{1}{3}, \frac{1}{2}] \\ 0.4 & \text{if } w(x) > \frac{1}{2} \end{cases}$$

Lemma 5. *The first-fit bin assignment step of our algorithm always succeeds, and with the choice of parameters we defined, Algorithm 6 is $\alpha(k) := \frac{2\lceil 1.7k+1.6 \rceil}{k}$-competitive.*

Proof. First, we observe that $\mathsf{IP} \geqslant \mathsf{OPT}$. Indeed, consider an optimal assignment of items to knapsacks. Lemma 1 in [GGJY76] shows that the adjusted weights in each knapsack is at most 1.7, so OPT is a feasible solution to the integer program. Lemma 5 in [GGJY76] shows that if t is the large index used by the bin assignment step in our algorithm, and I is the set of admitted items, then:

$$w'(I) > t - 1 + \sum_{i=1}^{t} \max(0, w'(B_i) - 1).$$

This establishes a bound on the total number of virtual knapsacks required as $t < w'(I)+1 \leqslant 1.7k+2.6$, or $t \leqslant \lceil 1.7k + 2.6 \rceil - 1 = \lceil 1.7k + 1.6 \rceil = f(k)$. The result then follows immediately from lemma 1. □

This completes proof of theorem 6. We note that other choices of parameters are possible in our algorithm, using for instance online bin packing algorithms from the Harmonic family [LL85].

8 Conclusion

We broaden the reach of single-sample prophet inequalities to knapsack and packing problems, obtaining tight and close-to-optimal competitive guarantees. Our results notably include a tight, 2-competitive algorithm for online fractional knapsack, a result that mirrors the celebrated result of [RWW19]: for fractional knapsack, just as for the classic single-choice problem, the optimal competitive ratio of 2 achievable with full distributional information can equally be obtained when prior knowledge is restricted to just a single sample.

For integral knapsack, our single-sample guarantee of 4 is only slightly worse than the state of the art with full distribution knowledge guarantee of 3.135 in [JMZ22], motivating the question of whether there actually exists a gap between the optimal competitive guarantees in the integral case.

As is common in knapsack settings, the loss in approximation when moving to the integral setting is very clearly, as our lower bound demonstrates, necessary and correlated with the existence of large items. Finally, we were able to also derive constant-factor SSPI for other packing type problems with richer structure, further outlining the power of the paradigm for such settings and motivating further study to investigate other variants.

Our guarantees are obtained via simple, mostly threshold-based policies, and hold without the assumption of large capacity commonly made in other work on online selection or packing problems with samples. The encouraging results of this work make it conceivable that just like in matching, combinatorial allocation and various matroid settings, most prophet inequalities for packing problems can be replaced by much more lightweight single-sample variants without much loss, considerably improving the algorithms' practical implementability.

9 Acknowledgements

The lower bound of 5/2 was constructed with assistance of a generative AI model (ChatGPT 5.5 Pro), via a conversation aiming to improve the lower bound beyond 2, and optimize the resulting example. The authors have verified this example, and assume responsibility for all content.

References

[AKL21] Susanne Albers, Arindam Khan, and Leon Ladewig. Improved online algorithms for knapsack and gap in the random order model. *Algorithmica*, 83(6):1750–1785, 2021.

[AKW14] Pablo Daniel Azar, Robert Kleinberg, and S. Matthew Weinberg. Prophet inequalities with limited information. In *Proceedings of SODA*, pages 1358–1377, 2014.

[ALSS22] Andreas Abels, Leon Ladewig, Kevin Schewior, and Moritz Stinzendörfer. Knapsack secretary through boosting. In *International Workshop on Approximation and Online Algorithms*, pages 61–81. Springer, 2022.

[BBD+17] János Balogh, József Békési, György Dósa, Leah Epstein, and Asaf Levin. A new and improved algorithm for online bin packing. *arXiv preprint arXiv:1707.01728*, 2017.

[BHK+26] Hans-Joachim Böckenhauer, Juraj Hromkovič, Dennis Komm, Peter Rossmanith, and Moritz Stocker. A survey of online knapsack problems. *Discrete Applied Mathematics*, 378:492–507, 2026.

[BIKK07] Moshe Babaioff, Nicole Immorlica, David Kempe, and Robert Kleinberg. A knapsack secretary problem with applications. In *International Workshop on Approximation Algorithms for Combinatorial Optimization*, pages 16–28. Springer, 2007.

[CDF+22] Constantine Caramanis, Paul Dütting, Matthew Faw, Federico Fusco, Philip Lazos, Stefano Leonardi, Orestis Papadigenopoulos, Emmanouil Pountourakis, and Rebecca Reiffenhäuser. Single-sample prophet inequalities via greedy-ordered selection. In *Proceedings of SODA*, 2022.

[CFH+19] Jose Correa, Patricio Foncea, Ruben Hoeksma, Tim Oosterwijk, and Tjark Vredeveld. Recent developments in prophet inequalities. *ACM SIGecom Exchanges*, 17(1):61–70, 2019.

[CGKM20] Shuchi Chawla, Kira Goldner, Anna R Karlin, and J Benjamin Miller. Non-adaptive matroid prophet inequalities. *arXiv preprint arXiv:2011.09406*, 2020.

[CHMS10] Shuchi Chawla, Jason D Hartline, David L Malec, and Balasubramanian Sivan. Multi-parameter mechanism design and sequential posted pricing. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 311–320, 2010.

[CK05] Chandra Chekuri and Sanjeev Khanna. A polynomial time approximation scheme for the multiple knapsack problem. *SIAM Journal on Computing*, 35(3):713–728, 2005.

[DFKL20] Paul Dutting, Michal Feldman, Thomas Kesselheim, and Brendan Lucier. Prophet inequalities made easy: Stochastic optimization by pricing nonstochastic inputs. *SIAM Journal on Computing*, 49(3):540–582, 2020.

[DGV05] Brian C Dean, Michel X Goemans, and Jan Vondrák. Adaptivity and approximation for stochastic packing problems. In *SODA*, volume 5, pages 395–404, 2005.

[DKL+24] Paul Dütting, Thomas Kesselheim, Brendan Lucier, Rebecca Reiffenhäuser, and Sahil Singla. Online combinatorial allocations and auctions with few samples. In *2024 IEEE 65th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1231–1250, 2024.

[DS13] György Dósa and Jirí Sgall. First fit bin packing: A tight analysis. In *30th International symposium on theoretical aspects of computer science (STACS 2013)*, pages 538–549. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2013.

[Dyn63] Evgenii Borisovich Dynkin. The optimum choice of the instant for stopping a markov process. *Soviet Mathematics*, 4:627–629, 1963.

[Ezr26] Tomer Ezra. Prophet inequality from samples: Is the more the merrier? In *Proceedings of the 2026 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2099–2112. SIAM, 2026.

[FGK71] C. M. Fortuin, J. Ginibre, and P. W. Kasteleyn. Correlation inequalities on some partially ordered sets. *Communications in Mathematical Physics*, 22(2):89 – 103, 1971.

[FSZ21] Moran Feldman, Ola Svensson, and Rico Zenklusen. Online contention resolution schemes with applications to bayesian selection problems. *SIAM Journal on Computing*, 50(2):255–300, 2021.

[GGJY76] Michael R Garey, Ronald L Graham, David S Johnson, and Andrew Chi-Chih Yao. Resource constrained scheduling as generalized bin packing. *Journal of Combinatorial Theory, Series A*, 21(3):257–298, 1976.

[GSW25] Rohan Ghuge, Sahil Singla, and Yifan Wang. Single-sample and robust online resource allocation. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*, pages 1442–1453, 2025.

[HK92] Theodore P Hill and Robert P Kertz. A survey of prophet inequalities in optimal stopping theory. *Contemporary Mathematics*, 125(1):191, 1992.

[JMZ22] Jiashuo Jiang, Will Ma, and Jiawei Zhang. Tight guarantees for multi-unit prophet inequalities and online stochastic knapsack. In *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1221–1246. SIAM, 2022.

[Kel99] Hans Kellerer. A polynomial time approximation scheme for the multiple knapsack problem. In *International Workshop on Randomization and Approximation Techniques in Computer Science*, pages 51–62. Springer, 1999.

[KK25] Max Klimm and Martin Knaack. Generalized assignment and knapsack problems in the random-order model. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 341–354. Springer, 2025.

[KNR22] Haim Kaplan, David Naori, and Danny Raz. Online weighted matching with a sample. In *Proceedings of SODA 2022*, pages 1247–1272, 2022.

[KS77] Ulrich Krengel and Louis Sucheston. Semiamarts and finite values. *Bulletin of the American Mathematical Society*, 83(4):745–747, 1977.

[KTRV14] Thomas Kesselheim, Andreas Tönnis, Klaus Radke, and Berthold Vöcking. Primal beats dual on online packing lps in the random-order model. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 303–312, 2014.

[KW12] Robert Kleinberg and Seth Matthew Weinberg. Matroid prophet inequalities. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 123–136, 2012.

[LL85] Chan C Lee and Der-Tsai Lee. A simple on-line bin-packing algorithm. *Journal of the ACM (JACM)*, 32(3):562–572, 1985.

[Luc17] Brendan Lucier. An economic view of prophet inequalities. *ACM SIGecom Exchanges*, 16(1):24–47, 2017.

[NW26] Pranav Nuti and Peter Westbrook. Static pricing for single sample multi-unit prophet inequalities. In *2026 SIAM Symposium on Simplicity in Algorithms (SOSA)*, pages 127–141. SIAM, 2026.

[Par25] Janusz Partyka. Online stochastic knapsack problems with a single sample. https://web.archive.org/web/20260430134604/https://dspace.uba.uva.nl/server/api/core/bitstreams/8b31c59a-22a0-40ce-a500-5015689aedc6/content, 2025. Accessed 2026-04-30.

[PS25] Kanstantsin Pashkovich and Alice Sayutina. Single sample prophet inequality for uniform matroids of rank 2. *Operations Research Letters*, 60:107257, 2025.

[PS26] Kanstantsin Pashkovich and Alice Sayutina. Non-adaptive prophet inequalities for minor-closed classes of matroids. *Discrete Applied Mathematics*, 383:26–43, 2026.

[RWW19] Aviad Rubinstein, Jack Z Wang, and S Matthew Weinberg. Optimal single-choice prophet inequalities from samples. *arXiv preprint arXiv:1911.07945*, 2019.

[SC84] Ester Samuel-Cahn. Comparison of threshold stop rules and maximum for independent nonnegative random variables. *the Annals of Probability*, pages 1213–1216, 1984.

[Ull71] Jeffrey D Ullman. The performance of a memory allocation algorithm. *(No Title)*, 1971.

[YK23] Toru Yoshinaga and Yasushi Kawase. Size-stochastic knapsack online contention resolution schemes. *arXiv preprint arXiv:2305.08622*, 2023.