STOCHASTIC PROBING WITH INCREASING PRECISION

MARTIN HOEFER*, KEVIN SCHEWIOR†, AND DANIEL SCHMAND‡

Abstract. We consider a selection problem with stochastic probing. There is a set of items whose values are drawn from independent distributions. The distributions are known in advance. Each item can be *tested* repeatedly. Each test reduces the uncertainty about the realization of its value. We study a testing model, where the first test reveals if the realized value is smaller or larger than the c-quantile of the underlying distribution of some constant $c \in (0, 1)$. Subsequent tests allow to further narrow down the interval in which the realization is located. There is a limited number of possible tests, and our goal is to design near-optimal testing strategies that allow to maximize the expected value of the chosen item. We study both identical and non-identical distributions and develop polynomial-time algorithms with constant approximation factors in both scenarios.

Key words. Stochastic Probing, Testing, Optimal Stopping

MSC codes. 68W25, 68W40, 68T05, 62L15

1. Introduction.

In recent years, there has been a surge of interest in learning problems with probing. There is a set of n items, and each item has an independent distribution over its value. The goal of the learner is to select an item with a value as large as possible. In the standard model, the learner can *probe* a bounded number of k items. Upon probing an item, the learner sees its realized value.

A variety of applications are captured by this approach and its extensions. For example, in a hiring process, an "item" is a candidate. The application material implies a stochastic belief over the quality of each candidate. Probing corresponds to an interview of a candidate, and the capacity of the interviewer is limited to k interviews. The probing problem corresponds to the selection of candidates to be interviewed to optimize the value of the candidate that is hired eventually. Additional applications arise, for example, in online dating or kidney exchange. The problem is to probe pairs of agents for compatibility and eventually match the population to maximize some objective function, e.g., the number of compatible pairs or the overall quality of matches. Probing has further applications in domains like influence maximization or Bayesian mechanism design [4, 19].

Computing an optimal probing decision is a non-trivial task as each of the subsequent probing decisions may depend on the outcomes of previous probes. A standard technique to design optimal probing strategies is a dynamic-programming approach, which often turns out to be intractable. Beyond this, one commonly resorts to finding polynomial-time approximation algorithms (e.g., [6, 8, 10, 12, 27]).

In the vast majority of approaches studied in theoretical computer science and applied mathematics, probing reveals the *exact realization* of the underlying random variable; probing an item completely eradicates the uncertainty. In contrast, many applications give rise to probing problems in which we only obtain *some limited information* about the item. Consider for example an interviewer in a hiring process. Instead of interpreting an interview as a single probe that reveals all information, it is usually the case that the interviewer can ask questions or request information that

*Institute for Computer Science, Goethe University Frankfurt, Robert-Mayer-Strasse 11-15, 60325 Frankfurt/Main, Germany. mhoefer@em.uni-frankfurt.de.

†Department of Mathematics and Computer Science, University of Southern Denmark, Campusvej 55, 5230 Odense M, Denmark. kevs@sdu.dk.

‡Center for Industrial Mathematics, University of Bremen, Bibliothekstraße 5, 28359 Bremen, Germany. schmand@uni-bremen.de.

This manuscript is for review purposes only.

will partially reveal the qualifications of the respective candidate. More realistically, a question or exercise in an interview can be seen as a probe, but only by asking multiple questions of varying levels of difficulty the interviewer can eventually estimate the exact qualification of each candidate. As another example, consider a Bayesian single-item auction with posted prices. By setting a posted price, the auctioneer learns if the bidders have a value above or below that price, but that does not directly reveal the valuation of each bidder. Only repeated probing with different prices can reveal the exact value of each bidder.

In this paper, we introduce selection problems with *repeated testing*. In the beginning, nature makes a single independent draw for each item to determine the realized value. We can test an item, but the exact realized value stays unknown and each test only reveals limited information about the realized value. Subsequent testing can be used to obtain more and more fine-grained information about the same realized value. There are many ways to express this condition formally, i.e., how exactly the result of a test changes the conditional distribution of the item. In our model, we take a simple and intuitive approach: The first test reveals if the realized value of an item is above or below the c-quantile of the distribution for some constant $c \in (0, 1)$. Each subsequent test reveals if the realized value of the item is in the c-quantile of the conditional distribution, where the condition is the binary feedback of previous tests.

EXAMPLE 1.1. Suppose we can perform $k = 3$ tests on $n = 2$ items and $c = 1/2$. Initially, the realized values of the items are drawn i.i.d. from the uniform distribution over the set $\{10, 20, 30, 40\}$.

W.l.o.g. we first test item 1. The result of the test is either positive (realization above the median) or negative (below the median). Suppose it is positive, then we know that the realized value of item 1 is either 30 or 40, both with probability $1/2$. Next, we test item 2. If the result for item 2 is negative, then the realized value of item 2 must be either 10 or 20, both with probability $1/2$. Hence, the optimum must be item 1. Instead, assume the result of the test for item 2 is positive, then the realized value of item 2 is either 30 or 40, both with probability $1/2$. We apply the third test again to item 1. If the result is positive, it is clear that the realization of item 1 is 40, and item 1 is an optimum. Otherwise, the realization is 30, and then item 2 is an optimum.

Interestingly, by repeating the analysis for the case when the result of the first test on item 1 is negative, we see that we can always identify the item with the best realization. Note that we cannot achieve this with $k \leq 2$ tests. ■

Before we discuss our results, let us formally introduce the model.

1.1. Testing. We are given a set $N = \{1, \dots, n\}$ of items and a test capacity $k \in \mathbb{N}$. The value v_i of each item $i \in N$ is non-negative, drawn independently from a known distribution D_i over $\mathbb{R}_+$, and unknown upfront. A *testing algorithm* can perform up to k tests. Each test is performed on one of the items. In contrast to most of the related literature, we assume that a test does not *reveal* the exact realization of the item's value. Instead, each test results in an improved estimation of the realization.

For simplicity, let us first assume a continuous distribution. Let $D_i^{-1}(q)$ be the smallest value such that $\mathbf{Pr}\left[v_i < D_i^{-1}(q)\right] = q$. We call $D_i^{-1}(q)$ the q-quantile of D_i. We assume that the first test shows whether the value is above or strictly below the c-quantile of the distribution for some constant $c \in (0, 1)$. That test is called

This manuscript is for review purposes only.

positive if $v_i \geq D_i^{-1}(c)$, and negative otherwise. Given this result, the conditional distribution of the item can then be tested for the new c-quantile in the same manner. In terms of the original distribution, if the first test was positive, the next test reveals if $v_i \geq D_i^{-1}(c + c(1-c))$ or $v_i \in [D_i^{-1}(c), D_i^{-1}(c + c(1-c)))$; if the first test was negative, the next test reveals if $v_i \in [D_i^{-1}(c^2), D_i^{-1}(c))$ or $v_i < D_i^{-1}(c^2)$. For the sake of exposition, we will usually first restrict attention to $c = 1/2$ (i.e., median tests) and then outline how to generalize our algorithms and analysis to general constants $c \in (0, 1)$.

In this way, repeated testing leads to an improved estimation of the realized value – each subsequent test informs the algorithm whether the value is above (positive test result) or strictly below (negative test result) the c-quantile of the conditional distribution, where the condition is on the outcomes of all previous tests on that item. The algorithm can perform k tests in total. It can choose the next item to be tested adaptively. In the end, it selects one item. The goal is to maximize the value of the selected item.

To aid the discussion of computational complexity, distributions are discrete and given in explicit representation. For applying the tests for such distributions, we assume that ties are broken consistently, e.g., by initially drawing a random number $x_i \in [0, 1]$ for each item i, extending D_i to a continuous distribution over tuples (v_i, x_i), and using a lexicographic comparison for tuples (v_i, x_i).

We provide algorithms that are polynomial-time in the input, where the input is given by the n discrete distributions in explicit representation and k. An algorithm can perform k tests on the items. Each test is executed via an oracle call that takes constant time.

1.2. Testing versus Probing. Our goal in this paper is to identify provably good testing algorithms. More fundamentally, our main interest is to relate the testing model to the standard probing model, where each of the k probes completely reveals the realization. How much value is lost due to the restriction that we only have access to repeated quantile-tests on the conditional distributions instead of immediate revelation of values? What is the *cost of testing instead of revealing*? Arguably, it is not obvious that this cost is small, for several reasons.

1. In standard probing, one can rather easily obtain a constant-factor approximation using *non-adaptive* algorithms that do not adjust the probing decisions to the revealed realizations, i.e., the *adaptivity gap* is constant. In contrast, good testing algorithms must necessarily be adaptive – we show in Section 2.1 that the adaptivity gap in testing is $\Theta(\log \min(n, k))$ even for i.i.d. items.
2. In the testing model, we are not aware of any direct application of adaptive submodularity [17], which guarantees that an adaptive version of the standard greedy algorithm yields a constant-factor approximation. Indeed, there are several natural algorithmic ideas (including the standard greedy algorithm) that fail to provide a constant-factor approximation, both with respect to an optimal strategy in standard probing as well as the optimal strategy in the testing model.

For details on point (1), see Section 2.1 below. We elaborate on point (2) in the following example.

EXAMPLE 1.2. Consider the following instance with n items and again $c = 1/2$. For simplicity, we describe the example using discrete distributions, but it is easy to adjust our observations to atomless distributions. For each of the *safe items*

This manuscript is for review purposes only.

$i = 1, \ldots, n/2$, the value is $v_i = 4$ independently with probability $1/n$ and $v_i = 3$ otherwise. The remaining items $i = n/2 + 1, \ldots, n$ are the *risky items* – their value is $v_i = n$ independently with probability $1/n$ and $v_i = 0$ otherwise. The number of tests or probes is $k = n/2$.

In standard probing, we can apply the k probes to the risky items to see their value. Consequently, the expected value is at least

$$\left(1 - (1 - 1/n)^{n/2}\right) n \geq \left(1 - \frac{1}{\sqrt{e}}\right) n \,.$$

For the testing scenario, we can apply k tests. Clearly, one objective is to obtain information about as many items as possible. Moreover, once we finished testing, it is optimal to pick an item that has the highest conditional expectation. As such, we want to test items repeatedly to increase the best conditional expectation. This motivates natural algorithmic approaches:

(a) Choose k different items (possibly adaptively) and test each item exactly once.
(b) For each test, pick an item with the currently highest conditional expectation, for which the value has not been fully determined.
(c) For each test, pick an item to maximize the expected marginal increase in the highest conditional expectation.

For algorithm (a), observe that both applying a single test to a risky or safe item rises the conditional expectation to at most 4. As such, algorithm (a) does not obtain an expected value above 4, independently of the choice of the items to test.

For algorithm (b), observe that initially the conditional expectation of every risky item is 1 and the one of every safe item is at least 3. Hence, the conditional expectation of every safe item is larger than the one of every risky item. It is easy to see that this invariant remains true throughout the algorithm. Hence, algorithm (b) only tests safe items. It eventually decides to pick a safe item, which has a value of at most 4.

For algorithm (c), if we test a risky item, then as observed above, the conditional expectation of the tested item rises to 2 with probability $1/2$, and it drops to 0 with probability $1/2$. As such, the first test on a risky item never increases the highest conditional expectation. Instead, the algorithm will test only safe items. Initially, the expected value of every safe item is $3 + 1/n$ for all $i \geq 1$. The conditional expectation of a safe item rises as long as the tests are positive. If the test is negative, the expectation drops to 3. As such, the algorithm will exclusively test safe items. It eventually decides to pick a safe item, which has value at most 4.

This shows that the value obtained by all these algorithms is only a $O(1/n)$-fraction of the value that can be obtained with k probes in the standard probing model. Our main result in this paper are testing algorithms that allow a constant-factor approximation to the value obtained in the standard probing model. ■

1.3. Contribution and Outline. In this paper, we provide two testing algorithms, one for identically distributed items and one for non-identical distributions, both running in polynomial time. We prove that they provide *constant approximation ratios*, which hold even with respect to the expected value of the best strategy in the standard probing model. In contrast to the approaches in the example above, our algorithms carefully choose the correct number of items to be tested. On the one hand, we need a sufficiently large set of items to be tested while, on the other hand, a sufficient (expected) number of tests must be available for each item to guarantee a small approximation ratio. Maybe surprisingly, striking a good balance between these

This manuscript is for review purposes only.

conflicting objectives is indeed possible.

Our algorithms are inherently adaptive. Indeed, we show that non-adaptive algorithms can only obtain an approximation ratio of $\Omega(\log \min(n, k))$ w.r.t. the expected value of the best testing strategy. As such, in contrast to probing, there is a *non-constant adaptivity gap*. We adjust our algorithms and obtain non-adaptive variants that yield asymptotically tight upper bounds on the adaptivity gap.

Our algorithms are conceptually different than the adaptive greedy procedures considered in the example above. They can be interpreted to consider items sequentially – we apply tests to the item under consideration until the item is accepted or discarded, and then the next item is tested (if tests remain). Therefore, the algorithms and analyses naturally extend to the sequential variant of the problem, where all tests on a single item must be applied consecutively, and the order of items for testing is externally given. For this variant, we also provide an efficient algorithm to compute the optimal testing strategy based on a dynamic program.

A preliminary version of the present paper was published in the proceedings of the 30th International Joint Conference on Artificial Intelligence (IJCAI) [23]. The present version extends the extended abstract by tight results on adaptivity gaps and general quantile tests.

Outline. After a discussion of related work in the subsequent Section 1.4, we describe in Section 2 our algorithm and the analysis for the case of independent and identically distributed (i.i.d.) items with $D_i = D_j = D$ for all $i, j \in N$. In Section 2.1 we bound the adaptivity gap for i.i.d. distributions to $\Theta(\log \min(n, k))$.

Our algorithm for the general case is considered in Section 3. In Section 3.1 we bound the adaptivity gap for the general case to $\Theta(\log \min(n, k))$. In Section 4 we consider the sequential testing problem, where items have to be tested sequentially in a given order. We conclude in Section 5.

1.4. Related Work. Stochastic probing problems in which probing eradicates all uncertainty about the tested item have been extensively studied. A prominent line of work [1, 5, 19–21] is concerned with fairly general models in which an—according to some given downward-closed set system—feasible set of (often Bernoulli) variables can be adaptively probed. When probing is done, a set of items that is feasible according to another given downward-closed set system can be selected, and the obtained value is an (e.g., submodular) function of the selected items. The goal is typically to develop algorithms that approximate the best strategy and whose guarantees are parameterized by the respective instance, e.g., parameters of the set systems corresponding to the constraints [6, 8, 10, 12, 27]. One approach to achieve constant-factor approximations is bounding both the adaptivity gap and the approximation factor of some non-adaptive algorithm by a constant; see, e.g, [20]. In this light, our approximation results, which are based on algorithms that work in the sequential setting, can be viewed as a bound on the "sequentiality gap" of our problem.

Instead of having to satisfy a hard constraint on the set of items that can be probed, in the *Pandora's box* problem one is charged for probing any of the items (for which the inherent values are typically independently distributed) [29]. The goal is to maximize the expected difference between the value of the chosen item and the probing cost. While in the standard model the picked item must be a previously probed item, in [7] any single item can be picked, but again probing eradicates all uncertainty. In a Markovian model [18], each probe only advances a Markov chain associated with the respective item by a single step. This is a model with limited information, but in contrast to our model an item may only be picked once the Markov chain has reached

This manuscript is for review purposes only.

a terminal state, i.e., once *all* uncertainty has been eradicated.

Some of these models have been generalized to variants, in which multiple items can be chosen; see, e.g., [28]. In the standard model, an optimal algorithm is known; more generally, one often resorts to approximation algorithms, sometimes even in the form of a PTAS [14].

The prophet-inequality setting [25] is different in that the values of all items are revealed eventually and there is no probing cost. In the classic version, items are revealed in an adversarial order, and a single item can only be picked at the time of its revelation. Then, the best strategy can be computed via a simple dynamic program, but the challenge is typically to compare the performance with that of an all-knowing prophet. When the order of revelation can be chosen, computing the best strategy becomes less tractable [2]. We also refer to surveys on this topic [11, 26].

Let us emphasize that our problem is quite different from multi-armed bandit models (e.g., [15]), in which typically actions have random payoffs from *unknown distributions*, from which samples are *repeatedly drawn and revealed*. In contrast, here each value for an item comes from a *known distribution*, is sampled only *once in the beginning* and only *revealed gradually* (upon testing). This setting calls for analyses different from the "regret"-style analysis typically applied for multi-armed bandit models.

2. Identical Distributions.

The main result in this section is our algorithm $\mathrm{ALG}_{\mathrm{iid}}$, which has a constant approximation ratio for identical distributions. The algorithm only depends on the test results and *uses no additional information about the distribution*. It is simple and achieves a good constant approximation guarantee, even with respect to the optimum in the standard probing model, where each test reveals the realization. We first discuss it in the setting of median tests, i.e., $c = 1/2$.

Algorithm $\mathbf{ALG_{iid}}$. Let $k' = 2^{\lceil \log_2 \min\{n,k+1\} \rceil}$ the smallest power of 2 that is larger or equal to $\min\{n, k+1\}$. We use the short notation $\delta_q = D^{-1}(q)$. Our algorithm $\mathrm{ALG}_{\mathrm{iid}}$ performs tests on the items sequentially. For each item i, it repeatedly tests the item until it is clear whether it's value v_i is larger or equal to $\delta_{(k'-1)/k'}$ or not, i.e., until there are $\log_2(k')$ positive tests in a row or until there is a single negative test. If $v_i \geq \delta_{(k'-1)/k'}$, we call this item a *good item*. In this case, the algorithm selects i and terminates. Otherwise, it continues by testing item $i+1$. If the algorithm fails to find a good item or runs out of tests, it selects a random item.

We slightly abuse notation and use $\mathrm{ALG}_{\mathrm{iid}}$ to denote our algorithm and $\mathbf{E}[\mathrm{ALG}_{\mathrm{iid}}]$ for the expected value of the chosen item. Our guarantee will relate this to the expected value of $\mathrm{ProbeOPT}_{k+1}$, the value obtained in the standard probing model by seeing the *exact* realization of the first $k+1$ items and selecting the one with the best realization. Instead of probing k items and then possibly taking an (unprobed) item $k+1$ (in case $k < n$ and all observed realizations are below the expectation of D), we allow $\mathrm{ProbeOPT}_{k+1}$ to also reveal the realization of item $k+1$ and then select the best realization from the $k+1$ probed items. Clearly, observing exact realizations and the additional probe imply that $\mathbf{E}[\mathrm{ProbeOPT}_{k+1}]$ upper bounds the value achievable by any algorithm in the testing scenario with k tests. Our main result in this section is the following.

THEOREM 2.1. $\mathrm{ALG}_{\mathrm{iid}}$ *runs in polynomial time and obtains a value of*

$$\mathbf{E}[\mathrm{ALG}_{\mathrm{iid}}] \geq \left(1 - \frac{1}{\sqrt[4]{\mathrm{e}}} - o(1)\right) \cdot \mathbf{E}[\mathrm{ProbeOPT}_{k+1}] \quad ,$$

where the asymptotics is in $\min(n, k)$.

This manuscript is for review purposes only.

Proof. We first assume $k < n$ and discuss the case $k \geq n$ below. In the subsequent Lemma 2.2, we prove a lower bound on the probability that ALG_{iid} finds a good item.

We start by observing that the expected value of any good item is at least $\mathbf{E}\left[\text{ProbeOPT}_{k+1}\right]$: Clearly, in ProbeOPT_{k+1} we have probability $1/(k+1)$ to select each of the first $k+1$ items. Under the condition that the probability of selecting an item is $1/(k+1)$, by stochastic dominance, the largest-possible expectation of the item's value is $\mathbf{E}\left[v_i \mid v_i \geq \delta_{k/(k+1)}\right]$. Additionally, $\mathbf{E}\left[v_i \mid v_i \geq \delta_x\right]$ is increasing in x and $k' \geq k+1$. Hence, $\mathbf{E}\left[\text{ProbeOPT}_{k+1}\right] \leq \mathbf{E}\left[v_i \mid v_i \geq \delta_{(k'-1)/k'}\right]$, the expected value of a good item.

By Lemma 2.2 below, we can conclude that the algorithm finds and selects a good item with probability at least

$$\alpha = 1 - \frac{1 - \frac{1}{2k'}}{1 - \frac{\log_2(k')}{k'}} \cdot \frac{1}{\sqrt{\mathrm{e}^{\frac{k+1}{k'+1}}}} - \frac{1}{2^{k-1}3} \geq 1 - \frac{1}{\sqrt[4]{\mathrm{e}}} - o(1),$$

where the asymptotics is in $k = \min(n, k)$. Since a good item has expected value of at least $\mathbf{E}\left[\text{ProbeOPT}_{k+1}\right]$ the approximation factor is at least α.

Finally, let us briefly discuss the case $k \geq n-1$. We can restrict ALG_{iid} to $n-1$ tests and apply the same analysis, where $n-1$ replaces k. On the other hand, clearly, $\mathbf{E}\left[\text{ProbeOPT}_{k+1}\right] = \mathbf{E}\left[\text{ProbeOPT}_{n}\right]$ for every $k \geq n-1$, since $k+1 \geq n$ probes are sufficient to reveal all values of all n items. ◻

LEMMA 2.2. *The probability that* ALG_{iid} *runs out of tests before finding a good item can be upper bounded by*

$$\frac{1 - \frac{1}{2k'}}{1 - \frac{\log_2(k')}{k'}} \cdot \frac{1}{\sqrt{\mathrm{e}^{\frac{k+1}{k'+1}}}} + \frac{1}{2^{k-1}3} .$$

The proof of Lemma 2.2 is rather technical and deferred to the appendix. Instead, we discuss a simple argument that the lower bound α on the competitive ratio in the proof of Theorem 2.1 is a constant, i.e., with probability $\Omega(1)$ the algorithm selects a good item before running out of tests.

LEMMA 2.3. *The probability that* ALG_{iid} *finds a good item before running out of tests can be lower bounded by a constant.*

Proof. If n is constant, then so are k and k'. Then, there is a constant probability that the first item is good and identified by the first $\log_2 k'$ tests. For the rest of the proof, we therefore assume $n > k \geq 6$ and, hence, $k' \geq 8$. Then the probability that the first $\lfloor k/4 \rfloor$ items contain at least one good one is

$$1 - (1 - 1/k')^{\lfloor k/4 \rfloor} \geq 1 - (1 - 1/k')^{\lfloor k'/8 \rfloor} = 1 - (1 - 1/k')^{k'/8} \geq 1 - \frac{1}{\mathrm{e}^{1/8}} .$$

For the rest of the proof, we condition on the fact that there is a good item among the first $\lfloor k/4 \rfloor$ items, denoted by $F_{1..\lfloor k/4 \rfloor}$. We now upper bound the probability that we do not identify the first good item by 18/19. This happens when we have less than $\log_2 k'$ remaining tests upon arriving at the first good item. Thus, we bound the probability that we use more than $k - \log_2 k'$ tests before arriving at the first good

This manuscript is for review purposes only.

item. We use F_j to denote the event that item j is the first good item. Then,

$$\begin{aligned}
&\mathbf{Pr}\left[\text{less than } \log_2 k' \text{ tests remain for the first good item} \mid F_{1..\lfloor k/4\rfloor}\right]\\
&= \sum_{j=1}^{\lfloor k/4\rfloor} \mathbf{Pr}\left[F_j \wedge (\text{less than } \log_2 k' \text{ tests remain for } j) \mid F_{1..\lfloor k/4\rfloor}\right]\\
&= \sum_{j=1}^{\lfloor k/4\rfloor} \mathbf{Pr}\left[F_j \wedge (\text{more than } k-\log_2 k' \text{ tests used before } j) \mid F_{1..\lfloor k/4\rfloor}\right]\\
&= \sum_{j=1}^{\lfloor k/4\rfloor} \mathbf{Pr}\left[F_j \mid F_{1..\lfloor k/4\rfloor}\right] \cdot\\
&\qquad\qquad \mathbf{Pr}\left[\text{more than } k-\log_2 k' \text{ tests used before } j \mid F_j \cap F_{1..\lfloor k/4\rfloor}\right]\\
&= \sum_{j=1}^{\lfloor k/4\rfloor} \mathbf{Pr}\left[F_j \mid F_{1..\lfloor k/4\rfloor}\right] \mathbf{Pr}\left[\text{more than } k-\log_2 k' \text{ tests used before } j \mid F_j\right] .
\end{aligned}$$

Consider the event of using more than $k-\log_2 k'$ tests on the bad items $\{1,\dots,j-1\}$. It has the same probability as the following event: In an infinite stream of bad items, for $k-\log_2 k'$ tests we see less than $j-1$ negative test results, or equivalently, more than $k-\log_2 k'-j+1$ positive test results. We use the random variable X for the number of positive test results and obtain

$$\begin{aligned}
&\sum_{j=1}^{\lfloor k/4\rfloor} \mathbf{Pr}\left[F_j \mid F_{1..\lfloor k/4\rfloor}\right] \cdot \mathbf{Pr}\left[\text{more than } k-\log_2 k' \text{ tests used before } j \mid F_j\right]\\
&= \sum_{j=1}^{\lfloor k/4\rfloor} \mathbf{Pr}\left[F_j \mid F_{1..\lfloor k/4\rfloor}\right] \cdot \mathbf{Pr}\left[X > k-\log_2 k' - j + 1\right]\\
&\le \sum_{j=1}^{\lfloor k/4\rfloor} \mathbf{Pr}\left[F_j \mid F_{1..\lfloor k/4\rfloor}\right] \cdot \mathbf{Pr}\left[X > k-\log_2 k' - \lfloor k/4\rfloor + 1\right]\\
&= \mathbf{Pr}\left[X > k-\log_2 k' - \lfloor k/4\rfloor + 1\right] \le \frac{\mathbf{E}[X]}{k-\log_2 k' - \lfloor \frac{k}{4}\rfloor + 1} ,
\end{aligned}$$

where the last inequality is due to Markov's inequality. Note that whenever we test a bad item, the probability of a positive test is strictly less than $1/2$. We obtain

$$\begin{aligned}
\frac{\mathbf{E}[X]}{k-\log_2 k' - \lfloor \frac{k}{4}\rfloor + 1} &< \frac{\frac{1}{2}(k-\log_2 k')}{k-\lfloor \frac{k}{4}\rfloor - \log_2 k' + 1}\\
&\le \frac{\frac{1}{2}(k-\log_2 k')}{\frac{3}{4}(k-\log_2 k') - \frac{1}{4}\log_2 k' + 1}\\
&\le \frac{\frac{1}{2}(k-\log_2 k')}{\frac{3}{4}(k-\log_2 k') - \frac{2}{9}(k-\log_2 k')} = \frac{18}{19} ,
\end{aligned}$$

where the last inequality follows because $\log_2 k' - 1 \le \frac{8}{9}(k-\log_2(k'))$ for $k \ge 6$.

Hence, conditioned on $F_{1..\lfloor k/4\rfloor}$, the probability that we fail to identify the first good item is at most $18/19$, so with probability at least $1/19$, we have enough tests to identify it. Overall, by multiplying with the probability of $F_{1..\lfloor k/4\rfloor}$, we get that a good item is found with probability at least $(1-\mathrm{e}^{-1/8})/19 \in \Omega(1)$. ∎

This manuscript is for review purposes only.

Testing for a c-quantile. Our analysis can be extended rather generically to the case when each test reveals if the realization is above or below a c-quantile of the conditional distribution for an item, for any constant $c \in (0,1)$. Then, using

$$k' = \left(\frac{1}{1-c}\right)^{\lceil \log_{1/(1-c)} \min\{n,k+1\} \rceil},$$

we define a good item as one where the first $r = \log_{1/(1-c)}(k') = \lceil \log_{1/(1-c)}(\min\{n, k+1\}) \rceil$ tests are all positive. The probability that we get such an item can be bounded by generalizing Lemma 2.2 from $c = 1/2$ to $c \in (0,1)$. Then the probability to find a good item is at least

$$\alpha_c = 1 - \frac{1}{\mathrm{e}^{c(1-c)}} - o(1),$$

which bounds the approximation ratio of the algorithm. For a more detailed discussion see the appendix.

2.1. Adaptivity Gap. Note that $\mathrm{ALG}_{\mathrm{iid}}$ is inherently adaptive in choosing the next item to test. A popular approach in probing problems is to design simpler *non-adaptive* probing strategies. Notably, in standard probing there is a constant adaptivity gap – the expected values of optimal adaptive and non-adaptive algorithms differ by at most a constant factor.

Here we show that testing is different in the sense that the adaptivity gap is non-constant.

THEOREM 2.4. *The adaptivity gap for testing with identical distributions is in* $\Omega(\log \min\{k, n\})$.

Proof. Suppose there are $k = 2^j$ tests and $n \geq k$ items with a gold-nugget distribution, for an integer $j > 1$. In this distribution, we have $v_i = k$ with probability $1/k$ and $v_i = 0$ otherwise. It is easy to see that by probing k items, we obtain an expected value of $\Omega(k)$, which asymptotically is also obtained by ($\mathrm{ALG}_{\mathrm{iid}}$ and, hence) the best adaptive testing strategy.

Now consider any non-adaptive testing strategy. The strategy divides the k tests onto the items before seeing any result. We number the items by the number of tests in non-increasing order, i.e., item i receives k_i tests, where $k_1 \geq k_2 \geq \ldots \geq k_n$ and $\sum_{i=1}^{n} k_i = k$.

W.l.o.g. we apply at most $k_i \leq j = \log_2 k$ tests to any item i, since with this number of tests we exactly learn the realization of that item. Consider the items in order of the numbering. With probability $1/2^{k_1}$ all k_1 tests on item 1 are positive. Then this item has conditional expectation 2^{k_1}, which is highest possible among all items and gets selected. If any of the k_1 tests on item 1 is negative, the item has value 0, is discarded, and we consider the k_2 tests on item 2. With probability $1/2^{k_2}$ all of them are positive, and then item 2 has conditional expectation 2^{k_2}. This is highest possible among all items, and item 2 gets selected. Otherwise, item 2 has value 0, is not selected, and we consider the k_3 tests on item 3, etc. Overall, the expected value

This manuscript is for review purposes only.

of the policy is

$$\begin{aligned}
&\frac{1}{2^{k_1}} \cdot 2^{k_1} + \left(1 - \frac{1}{2^{k_1}}\right) \cdot \frac{1}{2^{k_2}} \cdot 2^{k_2} + \ldots + \prod_{i=1}^{n-1} \left(1 - \frac{1}{2^{k_i}}\right) \cdot \frac{1}{2^{k_n}} \cdot 2^{k_n} \\
&\quad = 1 + \sum_{\ell=1}^{n-1} \underbrace{\prod_{i=1}^{\ell} \left(1 - \frac{1}{2^{k_i}}\right)}_{=:\ g(k_1,\ldots,k_\ell)} = 1 + \sum_{\ell=1}^{n-1} g(k_1, \ldots, k_\ell).
\end{aligned}$$

To derive an upper bound, consider each $g(k_1, \ldots, k_\ell)$ separately. $g(k_1, \ldots, k_\ell)$ is non-decreasing and concave when viewed as a continuous function in any k_i, and the dependence on all k_i is symmetric. We have a constraint $\sum_{i=1}^{n} k_i \leq k$. As such, g attains a maximum when $k_1 = \ldots = k_\ell = k/\ell$:

$$g(k_1, \ldots, k_\ell) \leq \left(1 - \frac{1}{2^{k/\ell}}\right)^{\ell}.$$

It is easy to see that the right term strictly decreases for $\ell = 1, \ldots, k$ from $1 - 1/2^k$ to $(1/2)^k$. For $\ell \leq 2k/(\log_2 k)$, we overestimate the value of $\left(1 - \frac{1}{2^{k/\ell}}\right)^{\ell} \leq 1$. For $2k/(\log_2(k)) < \ell \leq k$ we see that

$$\left(1 - \frac{1}{2^{k/\ell}}\right)^{\ell} < \left(1 - \frac{1}{2^{\log_2(k)/2}}\right)^{2k/(\log_2(k))} = \left(\left(1 - \frac{1}{\sqrt{k}}\right)^{\sqrt{k}}\right)^{2\sqrt{k}/(\log_2(k))} = o(1/k).$$

Finally for all $\ell > k$, it must be that $k_\ell = 0$, since k_i are non-negative integers, so at most k of them can be positive. Hence, $\sum_{\ell=k+1}^{n-1} g(k_1, \ldots, k_\ell) = 0$.

Overall, we see that

$$1 + \sum_{\ell=1}^{n-1} g(k_1, \ldots, k_\ell) < 1 + \frac{2k}{\log_2(k)} + k \cdot o(1/k) = O\left(\frac{k}{\log_2 k}\right).$$

Hence, the adaptivity gap is $\Omega(\log k) = \Omega(\log \min\{n, k\})$. □

For an upper bound on the adaptivity gap, consider a non-adaptive variant of $\mathrm{ALG_{iid}}$. We simply pick $\lfloor k'/(\log_2(k')) \rfloor$ items and apply $\log_2(k')$ tests to each of these items. The probability that we see a good item is at least

$$\begin{aligned}
1 - \left(1 - \frac{1}{2^{\log_2 k'}}\right)^{\lfloor \frac{k'}{\log_2 k'} \rfloor} &= 1 - \left(1 - \frac{1}{k'}\right)^{\lfloor \frac{k'}{\log_2 k'} \rfloor} \\
&= - \sum_{\ell=1}^{\lfloor k'/\log_2 k' \rfloor} \binom{\lfloor k'/\log_2 k' \rfloor}{\ell} \left(-\frac{1}{k'}\right)^{\ell} \\
&= \binom{\lfloor k'/\log_2 k' \rfloor}{1} \cdot \frac{1}{k'} - \binom{\lfloor k'/\log_2 k' \rfloor}{2} \cdot \left(\frac{1}{k'}\right)^2 \pm \ldots \\
&= \Omega\left(\frac{1}{\log k'}\right) - O\left(\frac{1}{(\log k')^2}\right)
\end{aligned}$$

Hence, the adaptivity gap is $O(\log k') = O(\log \min\{n, k\})$. We will slightly generalize this idea in Section 3.1 below. In Theorem 3.5 we obtain a similar upper bound even for general distributions.

This manuscript is for review purposes only.

3. General Distributions.

Our main result in this section is an algorithm that has a constant approximation ratio for non-identical, independent distributions D_i. As in the previous section, we first concentrate on the case $c = 1/2$, and we first assume $k < n$.

In the following, we first describe an (approximate) upper bound on the value that the optimum obtains. From this upper bound, we can derive a value p_i such that is sufficient to select item i with constant probability when it realizes above its $(1 - \Omega(p_i))$-quantile. We then discuss how to design an algorithm that achieves that. Eventually, we formally analyze the resulting algorithm.

We again relate the performance to $\mathbf{E}[\text{ProbeOPT}_\ell]$, the expected value of the optimal strategy in the standard probing model that can *adaptively* inspect $\ell \leq n$ of the items, learns their *exact* realization and then picks the best realization it has seen.

When adaptively inspecting the exact value of k items, we might eventually want to resort to an uninspected item with the maximum expected value (if all realizations are below that expectation). Instead, for ProbeOPT_{k+1} we can also learn the realization of this additional uninspected item and then pick the best one among the $k + 1$ items seen. This is clearly stronger than what we can achieve in the testing model with k tests. Our main result is to provide an algorithm with constant approximation w.r.t. $\mathbf{E}[\text{ProbeOPT}_{k+1}]$.

Again, this also implies an $\Omega(1)$-approximation for $k \geq n$, since $n - 1$ probes to suffice to achieve a $\Omega(1)$-approximation with respect to $\mathbf{E}[\text{ProbeOPT}_n]$, which always learns and selects the best item—a trivial upper bound on what can be achieved with any kind of testing. As such, we can run our strategy using only $n - 1$ tests (and ignoring the rest). For the remainder of the section, we therefore concentrate on the case $k < n$.

As a first step, we apply a reduction to concentrate on a smaller number of relevant items. We do so using the following result from the literature, rephrased for our needs.

THEOREM 3.1 (Theorem 2 in [4]). *There exists an algorithm that, given $k \in \mathbb{N}$, in polynomial time selects a subset $N_{k+1} \subseteq N$ of the items with $|N_{k+1}| = k + 1$ and*

$$\mathbf{E}\left[\max_{i \in N_{k+1}} v_i\right] \geq \left(1 - \frac{1}{\mathrm{e}}\right) \cdot \mathbf{E}[\text{ProbeOPT}_{k+1}].$$

In contrast to [4] we have direct access to the distributions. By inspecting their analysis, we see that this implies the stated approximation without reduction by an $\varepsilon > 0$.

Now given the subset N_{k+1}, we apply a further random sampling step—we pick a uniformly random subset $N' \subset N_{k+1}$ of k' items. Clearly, we sample the item with the best realization from N_{k+1} with probability $k'/(k + 1)$. Thus,

$$\mathbf{E}\left[\max_{i \in N'} v_i\right] \geq \frac{k'}{k+1} \cdot \mathbf{E}\left[\max_{i \in N_{k+1}} v_i\right] \quad .$$

We choose $k' := \lfloor k/10 \rfloor$ so that k' is smaller than $k+1$ by a large-enough constant factor in order to be able to perform enough tests on the items of N'. Also, since $k' \in \Omega(k)$, we get $\mathbf{E}[\max_{i \in N'} v_i] = \Omega(1) \cdot \mathbf{E}[\text{ProbeOPT}_{k+1}]$. For convenience, we renumber the items such that $N' = [k'] = \{1, \ldots, k'\}$.

Furthermore, we assume $k > k_0$ for a suitable constant ($k_0 = 50$ is sufficient), since our analysis relies on concentration bounds and we need to ensure $k' \in \mathbb{N}$.

This manuscript is for review purposes only.

Otherwise, for constant $k \leq k_0$, selecting an item with the best (a priori) expectation $\mathrm{ALG}_{\mathrm{gen}}$ trivially guarantees a constant-factor approximation.

It remains to achieve a constant approximation to $\mathbf{E}\left[\max_{i\in[k']} v_i\right]$ under the assumption $k > k_0$. Let $\mathcal{E}_i$ be the event that i has the largest value of all items in N'. Here, we break ties in order of lower item numbers. We can write

$$\mathbf{E}\left[\max_{i\in[k']} v_i\right] = \sum_{i=1}^{k'} \mathbf{Pr}\left[\mathcal{E}_i\right] \cdot \mathbf{E}\left[v_i \mid \mathcal{E}_i\right]. \tag{3.1}$$

In the following we will use p_i as shorthand for $\mathbf{Pr}\left[\mathcal{E}_i\right]$ for all $i \in [k']$. Given explicit representations of the discrete distributions D_i for items in $[k']$, the values p_i can be computed easily in polynomial time[1].

We try to pick each item $i \in [k']$ that realizes to any fixed value above the $(1 - \Omega(p_i))$-quantile with constant probability. Then, with (3.1) and a similar argument as for identical distributions, we indeed get an $\Omega(1)$-approximation. Our algorithm again operates sequentially over the items. It considers items $1, \ldots, k'$ in arbitrary order, say, in ascending order of their indices. Upon considering item i, it (approximately) checks if v_i realizes above the $1 - p_i$ quantile of D_i. If this check succeeds, it simply selects item i; otherwise it discards i and proceeds with the next item.

Assuming we could perform the check for the $1 - p_i$ quantile not only approximately but exactly in our model (say, using $\Theta(\log(1/p_i))$ tests), this algorithm would *not* obtain all realizations above the $1 - \Omega(p_i)$ quantile with constant probability for all i; indeed, we need a specific approximate check. First, p_1 may be arbitrarily close to 1. Then we are unable to guarantee to arrive at item 2 with a constant probability and thereby fail to select v_2 with constant probability when v_2 realizes to a value above the $1 - \Omega(p_2)$ quantile of D_2. Second, p_1 may be so small that $\Theta(-\log p_1)$ exceeds k, the number of available tests. Then we never select v_1.

We address both issues by defining

$$q_i = \frac{\max\{p_i, 1/k'\}}{8} \in \Omega(p_i)$$

and using q_i in place of p_i. Lifting values smaller than $1/k'$ to $1/k'$ can be seen as an idea borrowed from the setting of identical distributions. Dividing the resulting probability by 8 makes sure that there is a constant lower bound on the probability that for any given item i the algorithm eventually considers i. A similar idea is used in Bayesian mechanism design [3, 9] and LP-based probing algorithms [6].

To (approximately) check more easily if v_i realizes above $D_i^{-1}(1 - q_i)$, we round q_i to a power of 2 (with negative exponent). We define $\tilde{q}_i$ to be the largest power of 2 which is at most q_i. Having arrived at item i, our algorithm tests item i at most $-\log_2 \tilde{q}_i \in \mathbb{N}$ times. As soon as one of the tests is negative, we stop testing item i and continue with the next item; if all tests are positive, we select item i. This concludes the description of our algorithm, which we summarize as $\mathrm{ALG}_{\mathrm{gen}}$. For a formal and precise description, see Algorithm 3.1. Recall that the analysis for the case $k > n - 1$ follows from restricting attention to $\min(k, n)$ tests.

The main result is the following theorem. By slight misuse of notation, we use $\mathbf{E}\left[\mathrm{ALG}_{\mathrm{gen}}\right]$ to denote the expected value of the item selected by our algorithm.

[1] For each possible realization v_i of item i, compute the probability that item i has value v_i, all items $j = 1, \ldots, i-1$ have a realization $v_j < v_i$, and all items $j = i+1, \ldots, k'$ have a realization $v_j \leq v_i$. The product of these numbers is the probability that v_i constitutes the maximum of all realizations. p_i is the sum of probabilities computed for all realizations of item i.

This manuscript is for review purposes only.

Algorithm 3.1 $\mathrm{ALG}_{\mathrm{gen}}$ for General Distributions

Input : Distributions $D_1, \ldots, D_n$ over $\mathbb{R}_+$, $k \in \mathbb{N}$.
Output: The index of the picked item.

if $k \le k_0$, **return** $i \in \arg\max_{i \in [n]} \mathbf{E}[v_i]$.
Required tests: $k \leftarrow \min(k, n-1)$.
Select set N_{k+1} of items using Theorem 3.1.
$k' \leftarrow \lfloor k/10 \rfloor$.
Select set N' of k' items from N_{k+1} uniformly at random; w.l.o.g. $N' = [k']$.
for i **in** $[k']$**:**
  $\mathcal{E}_i$ is the event that $\arg\max_{i \in [k']} v_i$ is item i (breaking ties arbitrarily).
  $q_i \leftarrow \max\{\mathbf{Pr}[\mathcal{E}_i], 1/k'\}/8$.
  $\tilde{q}_i \leftarrow 2^{\lfloor -\log_2 q_i \rfloor}$.
  for j **in** $[-\log_2 \tilde{q}_i]$**:**
    if *test is available***:**
      test distribution D_i.
      if *negative test result***: break** inner loop.
      if $j = -\log_2 \tilde{q}_i$**: return** i.
return any $i \in [k']$

Theorem 3.2. $\mathrm{ALG}_{\mathrm{gen}}$ *runs in polynomial time and achieves an expected value of*

$$\mathbf{E}[\mathrm{ALG}_{\mathrm{gen}}] \ge \Omega(1) \cdot \mathbf{E}[\mathrm{ProbeOPT}_{k+1}] \ .$$

To prove this theorem, we first show the following lemma.

Lemma 3.3. *Suppose* $k > k_0$*. There is a constant* $r > 0$ *such that, for any* $i \in [k']$*, the probability that* $\mathrm{ALG}_{\mathrm{gen}}$ *arrives at item* i *with at least* $\log_2 k' + 4$ *unused tests is at least* r*.*

Proof. It suffices to consider the event that $\mathrm{ALG}_{\mathrm{gen}}$ arrives at the last item, i.e., item k', with $\log_2 k' + 4$ unused tests, called $\mathcal{F}$ in the following, and bound its probability from below by a constant. By the union bound, we can write

$$\mathbf{Pr}[\mathcal{F}] \ge 1 - \mathbf{Pr}[\mathcal{F}_1] - \mathbf{Pr}[\mathcal{F}_2]. \quad (3.2)$$

Here, $\mathcal{F}_1$ is the event that the algorithm picks any v_i prior to even considering $v_{k'}$. To define $\mathcal{F}_2$, we view the tests as independent, unbiased coins and realize *all* of them, even those that are potentially not used by the algorithm. Now $\mathcal{F}_2$ is the event that among the first $k-(\log_2 k'+4)$ tests, fewer than $k'-1$ have result 0. Indeed, whenever $\mathcal{F}$ does not occur, at least one of $\mathcal{F}_1$ and $\mathcal{F}_2$ occurs.

We first consider $\mathcal{F}_1$. Note that $\sum_{i \in [k']} p_i = 1$. Since $\max\{p_i, 1/k'\} \le p_i + 1/k'$ for all $i \in [k']$, it follows that $\sum_{i \in [k']} \max\{p_i, 1/k'\} \le 2$, so $\sum_{i \in [k']} q_i \le 1/4$ by definition of q_i. Then, using $\tilde{q}_i \le q_i$ for all $i \in [k']$, we have $\sum_{i \in [k']} \tilde{q}_i \le 1/4$.

Since the probability that we pick item i is at most $\tilde{q}_i$ (for that to happen, v_i has to realize above the $1-\tilde{q}_i$ quantile of D_i), again by the union bound, the probability that we pick *any item at all* is at most $1/4$. Therefore

$$\mathbf{Pr}[\mathcal{F}_1] \le \frac{1}{4}. \quad (3.3)$$

This manuscript is for review purposes only.

It remains to bound $\mathbf{Pr}[\mathcal{F}_2]$ from above and away from 3/4. Towards applying Markov's inequality define X to be the number of positive tests among the first $\lfloor k/2 \rfloor$ tests. Then X has expectation at most $k/4$. We get

$$\mathbf{Pr}[\mathcal{F}_2] \leq \mathbf{Pr}\left[X \geq \frac{4k}{10}\right] \leq \mathbf{Pr}\left[X \geq \left(1+\frac{3}{5}\right)\cdot \mathbf{E}[X]\right] \leq \frac{5}{8}, \tag{3.4}$$

where the first inequality we use follows using $k > k_0 = 50$: When $\mathcal{F}_2$ occurs, we have less than $k' \leq k/10$ tests with result 0 among the first $\lfloor k/2 \rfloor < k - (\log_2 k' + 4)$ tests, so $X \geq k/2 - k' \geq 4k/10$ follows. The second inequality follows by plugging in the upper bound on the expected value of X, and the last inequality follows from Markov's inequality (clearly, $X \geq 0$).

The claim follows from combining Inequalities (3.3) and (3.4) in (3.2). ∎

With this lemma at hand, we can prove the main theorem.

Proof of Theorem 3.2. First consider the case $k \leq k_0 = 50$. We denote the returned index by $i^\star \in \arg\max_{i\in[n]} \mathbf{E}[v_i]$. Here we overestimate $\mathbf{E}[\text{ProbeOPT}_{k+1}]$ by selecting all $k+1$ observed realizations and obtaining the sum of the values. For this objective, it is trivially optimal to select the set $I^\star_{k+1}$ which we define to be the set of $k+1$ items with highest expectation. Since ALG_{gen} selects the single item with highest expectation, it recovers at least

$$\mathbf{E}[v_{i^\star}] \geq \frac{1}{k+1} \cdot \sum_{i \in I^\star_{k+1}} \mathbf{E}[v_i] \geq \frac{1}{k_0+1} \cdot \mathbf{E}[\text{ProbeOPT}_{k+1}] \ ,$$

implying our claim.

Now consider the case $k > k_0 = 50$. By Lemma 3.3, there exists a constant $r > 0$ such that with probability at least r, for any given item i, the algorithm arrives at i with at least $-\log_2 \tilde{q}_i \leq \log_2 k' + 4$ unused tests. Hence,

$$\begin{aligned}
\mathbf{E}[\text{ALG}_{\text{gen}}] &\geq \sum_{i=1}^{k'} r \cdot \mathbf{Pr}\left[v_i \geq D_i^{-1}(1-\tilde{q}_i)\right] \cdot \mathbf{E}\left[v_i \mid v_i \geq D_i^{-1}(1-\tilde{q}_i)\right] \\
&= r \cdot \sum_{i=1}^{k'} \tilde{q}_i \cdot \mathbf{E}\left[v_i \mid v_i \geq D_i^{-1}(1-\tilde{q}_i)\right] \\
&\geq r \cdot \sum_{i=1}^{k'} \frac{p_i}{16} \cdot \mathbf{E}\left[v_i \mid v_i \geq D_i^{-1}\left(1-\frac{p_i}{16}\right)\right] \\
&\geq r \cdot \sum_{i=1}^{k'} \frac{1}{16} \cdot \mathbf{Pr}[\mathcal{E}_i] \cdot \mathbf{E}[v_i \mid \mathcal{E}_i] \quad = \quad \frac{r}{16} \cdot \mathbf{E}\left[\max_{i\in[k']} v_i\right].
\end{aligned} \tag{3.5}$$

In the first step, we use the independence of arriving at item i and its realization v_i. The second step uses the definition of D_i. The third step follows by monotonicity of $x \cdot \mathbf{E}\left[v_i \mid v_i \geq D_i^{-1}(1-x)\right]$ as a function of x and $\tilde{q}_i \geq q_i/2 \geq p_i/16$. In the fourth step, we use that $p_i = \mathbf{Pr}[\mathcal{E}_i]$ for the first part and stochastic dominance to compare the two expected values. The last step uses the definition of $\mathcal{E}_i$.

Recalling the discussion of Theorem 3.1 and the random sampling step, we observe

$$\mathbf{E}\left[\max_{i\in[k']} v_i\right] \geq \left(1-\frac{1}{\mathrm{e}}\right) \cdot \frac{k'}{k+1} \cdot \mathbf{E}[\text{ProbeOPT}_{k+1}]. \tag{3.6}$$

This manuscript is for review purposes only.

The ratio follows by combining (3.5) and (3.6) with

$$k' = \left\lfloor \frac{k}{10} \right\rfloor \geq \frac{k}{10} - 1 \geq \frac{1}{13}(1.3k - 13) \geq \frac{1}{13}(k+2)\ ,$$

as $k \geq 50$. The running time of $\mathrm{ALG}_{\mathrm{gen}}$ is dominated by applying the algorithm of [4] and computing the values $p_i = \mathbf{Pr}[\mathcal{E}_i]$. Both steps run in time polynomial in the input size. ∎

Testing for a c-quantile. When tests return whether the realization is above or below the c-quantile for some constant $c \in (0, 1)$ (instead of 1/2-quantile) of the conditional probability distribution, the same techniques can be used to obtain an $\Omega(1)$-approximation. We provide a sketch of the adjusted algorithm $\mathrm{ALG}'_{\mathrm{gen}}$ and how the arguments can be adjusted. We choose $k' := \lfloor c \cdot k/5 \rfloor$ and k_0 as a sufficiently large constant (discussed below). With this adjusted definition of k' and k_0, we apply the same steps as in $\mathrm{ALG}_{\mathrm{gen}}$ until line 5 of the algorithm. As in the $c = 1/2$ case, for every $i \in [k']$ we can define a quantile

$$q_i := \max\{p_i, 1/k'\} \cdot \frac{c}{4}\ .$$

Choosing $\tilde{q}_i$ to be the largest power of c with $\tilde{q}_i \leq q_i$, we get

(i) $\tilde{q}_i \geq c \cdot p_i \cdot c/4$ for all $i \in [k']$,

(ii) $\tilde{q}_i \geq c/k' \cdot c/4 =: L$ for all $i \in [k']$,

(iii) $\sum_{i \in [k']} \tilde{q}_i \leq c/2$.

We then apply lines 6–15 of $\mathrm{ALG}_{\mathrm{gen}}$ with this adjusted definition of $\tilde{q}_i$ and $-\log_{1/c} \tilde{q}_i$ instead of $-\log_2 \tilde{q}_i$ in lines 10 and 14. Consider the following more general version of Lemma 3.3.

LEMMA 3.4. *Suppose $k > k_0$. There is a constant r such that, for any $i \in [k']$, the probability that $\mathrm{ALG}'_{\mathrm{gen}}$ arrives at item i with at least $\lfloor \log_c L \rfloor$ unused tests is at least r.*

For the proof, we can use (iii) to bound $\mathbf{Pr}[\mathcal{F}_1]$ from above by $c/2$, where $\mathcal{F}_1$ is again the event that the algorithm picks any item before considering the final one. Similarly, $\mathcal{F}_2$ is again the event that the number of negative tests among the first $k - \lfloor \log_c L \rfloor$ tests is smaller than $k' - 1$. To bound $\mathbf{Pr}[\mathcal{F}_2]$ we define X to be the number of positive tests among the first $\lfloor k/2 \rfloor$ tests, so that X has expected value at most $(1 - c) \cdot k/2$. Similarly to the previous analysis, we can write

$$\text{(3.7)}\quad \mathbf{Pr}[\mathcal{F}_2] \leq \mathbf{Pr}\left[X \geq \frac{5k - 2ck}{10}\right] \leq \mathbf{Pr}\left[X \geq \frac{5 - 2c}{5 - 5c} \cdot \mathbf{E}[X]\right] \leq \frac{5 - 5c}{5 - 2c} < 1 - \frac{c}{2}.$$

Towards the choice of k_0, we assume it is large enough to exclude all (constantly many) small values of k for which $\lfloor k/2 \rfloor > k - \lfloor \log_c L \rfloor$. As such, we can assume $\lfloor k/2 \rfloor \leq k - \lfloor \log_c L \rfloor$, and the first inequality of (3.7) follows because then $\mathcal{F}_2$ only occurs if $X \geq k/2 - k' \geq (5k - 2ck)/10$. As before, the next step follows by the upper bound on $\mathbf{E}[X]$, the step after that using Markov's inequality, and the final step by simple calculus. The proof of the constant-factor approximation is then analoguous to that of Theorem 3.2, using (i) and Lemma 3.4.

3.1. Adaptivity Gap. Turning to the adaptivity gap, we show that a non-adaptive variant of $\mathrm{ALG}_{\mathrm{gen}}$ guarantees a logarithmic upper bound. The lower bound has been established for identical distributions in Theorem 2.4 above.

This manuscript is for review purposes only.

THEOREM 3.5. *The adaptivity gap for testing with general distributions is in* $\Theta(\log\min\{k,n\})$.

Proof. For the upper bound consider a non-adaptive variant of $\mathrm{ALG}_{\mathrm{gen}}$. In this variant, we apply the same steps until line 5 of Algorithm 3.1. Then in line 6, instead of sequentially searching through all items from $[k']$, we pick a random subset N'' of $\lfloor k'/\log_2(16k')\rfloor$ items from $[k']$. Using the definitions of $\mathcal{E}_i$, q_i and $\tilde{q}_i$ as given in lines 7–9 (using k' and $[k']$), we apply $-\log_2\tilde{q}_i$ tests to each item $i\in N''$. Whenever there is at least one item $i\in N''$ for which all $-\log_2\tilde{q}_i$ test are positive, we return such an item with smallest index.

First, let us argue that we have enough tests to execute this algorithm. By definition $\tilde{q}_i\geq q_i\geq 1/(16k')$, so $-\log_2\tilde{q}_i\leq\log_2(16k')$. Overall, the algorithm considers $\lfloor k'/\log_2(16k')\rfloor$ items and applies at most $\log_2(16k')$ tests to each item. In total, these sum to at most $k'\leq k$ tests.

Now consider the approximation ratio. Consider an instance and a given random draw of the values v_i. Suppose we execute both $\mathrm{ALG}_{\mathrm{gen}}$ and the non-adaptive variant. We couple the random choices in these executions in the sense that both algorithms choose the same sets N_{k+1} and N'. Then, if $\mathrm{ALG}_{\mathrm{gen}}$ returns any item i, this must be the item from $[k']$ with smallest index such that all $-\log_2\tilde{q}_i$ tests were positive. For the non-adaptive variant, this item is selected into N'' with probability

$$\frac{k'}{\lfloor k'/\log_2(16k')\rfloor}\in\Omega\left(\frac{1}{\log k'}\right),$$

and in that case also gets returned. Hence, for every item i returned by $\mathrm{ALG}_{\mathrm{gen}}$, the non-adaptive variant returns the same item with probability $\Omega(1/\log k')$. The expected value of the non-adaptive variant is therefore at least $\Omega(1/\log k')\cdot\mathbf{E}[\mathrm{ALG}_{\mathrm{gen}}]$. Finally, note that $\mathrm{ALG}_{\mathrm{gen}}$ has a constant approximation ratio and $k'=\Theta(\min(k,n))$. The theorem follows. ∎

4. Sequential Testing.

We consider a sequential scenario of the testing problem, in which tests for the same item must be conducted consecutively, and items must be tested in a given order. This restricts the algorithm and the optimal testing strategy in two ways.

First, if a test series for an item j is stopped, j cannot be tested anymore. This restriction is very natural in many practical applications such as the hiring process discussed above. Typically, a candidate cannot be interviewed again after the job interview is finished. Additional applications for this assumption include flat viewings, inspection of second hand articles, or test series with time consuming test setups.

Second, we restrict all testing to adhere to a fixed ordering of the items, i.e., the order of items, in which they can be tested, is given upfront. Note that this constraint has no bite for the i.i.d. scenario.

Interestingly, all our results from the previous sections directly carry over to the sequential testing problem. Both our algorithms test each item using a single consecutive test series and can be applied when given *any* fixed order of items.

OBSERVATION 4.1. *Algorithms* $\mathrm{ALG}_{\mathrm{iid}}$ *and* $\mathrm{ALG}_{\mathrm{gen}}$ *run in polynomial time and obtain constant approximation factors for the sequential testing problem.*

4.1. A Dynamic Program for Sequential Testing.

As the main result in this section, we show how to compute the optimal testing strategy in polynomial time.

THEOREM 4.2. *The optimal strategy in the sequential testing problem can be computed in polynomial time.*

This manuscript is for review purposes only.

For the proof, we denote the test results of k_i tests on some item $i \in [n]$ by a vector $R \in \{0,1\}^{k_i}$ where 0 and 1 correspond to negative and positive tests, respectively. Moreover, we use $D_{i,R}$ for the distribution of v_i conditioned on the test results R. For simplicity, we restrict to $c = 1/2$ in this section; a generalization to any $c \in (0,1)$ is straightforward.

Observe that, in any given state of the system, optimal testing and selection decisions can be made knowing the instance parameters as well as

(i) the first item i_{next} that one has not stopped testing (w.l.o.g. $i_{\text{next}} \le n$),

(ii) the conditional distribution $D_{i_{\text{next}},R}$ of the item tested last (if any; otherwise $D_{i_{\text{next}},R} := \emptyset$), where R are the results of the tests conducted on item i_{next},

(iii) the conditional distribution $D_{i^\star,R'}$ of a previously considered item (if any; otherwise $D_{i^\star,R'}$ is the distribution $\emptyset$ that has mass 1 on value 0) $i^\star$ that maximizes $\mathbf{E}[v_{i^\star} \mid R']$, where again R' are the results of the tests conducted on $i^\star$, and

(iv) the remaining number of tests.

Due to the fixed ordering of items, we do not need to keep track of the history of *all* previously tested items, and (iii) suffices. More formally, we define

$$\mathcal{D}_i := \{D_{i,R} \mid R \in \{0,1\}^{k_i}, k_i \in [k]\},$$

and each entry of our DP corresponds to a quadruple in

$$[n] \times \left(\{\emptyset\} \cup \bigcup_{i \in [n]} \mathcal{D}_i\right)^2 \times \{0,\dots,k\}, \tag{4.1}$$

corresponding to the four parameters described above.

One may be tempted to think that superpolynomial running time is required in the dynamic program because (ii) and (iii) depend on the outcomes of possibly $\omega(\log k)$ tests, leading to $2^{\omega(\log k)} = \omega(\text{poly}(k))$ different results of these tests and a seemingly superpolynomial cardinality of $\mathcal{D}_i$. The key observation, however, is that there is only a polynomial number of possibilities for $D_{i,R}$, for any item i after $O(k)$ tests with result R. This holds since distributions D_i are discrete and come in explicit representation. Recall that a distribution D is called degenerate if $|\text{supp}(D)| = 1$. For simplicity, we use supp to denote the *essential* support of a distribution, which ignores elements of measure 0.

Lemma 4.3. *Suppose item $i \in [n]$ has been tested $k_i \le k$ times. Then the distribution $D_{i,R}$ is non-degenerate for at most $|\text{supp}(D_i)| - 1$ many distinct $R \in \{0,1\}^{k_i}$.*

Proof. Let again $i \in [n]$, $k_i \le k$, and $R \in \{0,1\}^{k_i}$. Note that $D_{i,R}$ is uniquely defined through the *inverse* of its cumulative density function, denoted by $D_{i,R}^{-1} : [0,1] \to \mathbb{R}_+$. Furthermore note that

$$D_{i,R}^{-1}(x) = D_i^{-1}((\ell + x) \cdot 2^{-k_i}) \quad \forall x \in [0,1]$$

for $\ell \in [2^{k_i}]$, the number represented by R when interpreted as binary number. Hence, when D_i^{-1} is constant on the interval $I_R := [\ell \cdot 2^{-k_i}, (\ell+1) \cdot 2^{-k_i})$, then $D_{i,R}^{-1}$ is constant (up to possibly $\ell \cdot 2^{-k_i}$) on its entire domain, and therefore $D_{i,R}$ is degenerate. To see that this is the case for all but $|\text{supp}(D_i)| - 1$ many values of R, note that any two intervals $I_{R'}$ and $I_{R''}$ for $R', R'' \in \{0,1\}^{k_i}$ and $R' \ne R''$ are disjoint. Since D^{-1} is a step function with $|\text{supp}(D_i)|$ many steps, D_i is indeed constant on I_R for all but $|\text{supp}(D_i)| - 1$ values of R. ∎

This manuscript is for review purposes only.

Hence, we can count the number of conditional distributions $D_{i,R}$ after $k_i \leq k$ tests with results R as follows: If $D_{i,R}$ is degenerate, there are precisely $|\text{supp}(D_i)|$ different possibilities for $D_{i,R}$. If $D_{i,R}$ is not degenerate, there are precisely $k+1$ possibilities for k_i, and for each such possibility, there are at most $|\text{supp}(D_i)| - 1$ possibilities for $D_{i,R}$ by Lemma 4.3. Therefore $|\mathcal{D}_i| \in O(k \cdot |\text{supp}(v_i)|)$, which is polynomial in the input length. Thus the cardinality of the set in Equation (4.1) and, hence, the number of DP entries is bounded by a polynomial in the input length.

We now describe how to explicitly compute the DP entries, which are the expected values that can be achieved starting in the situation described by the respective quadruples. Towards this, consider a DP entry $\text{DP}(i_{\text{next}}, D_{i_{\text{next}},R}, D_{i^\star,R'}, k')$. We start by discussing base cases. If $k' = 0$, then no more tests can be conduted, so the strategy just picks the box with largest expected value conditioned on all test outcomes, i.e.,

$$\begin{aligned}\text{DP}(i_{\text{next}}, D_{i_{\text{next}},R}, &D_{i^\star,R'}, 0) := \\ &\max\left\{\mathbf{E}\left[v_{i_{\text{next}}} \mid R\right], \mathbf{E}\left[v_{i^\star} \mid R'\right], \max_{i \in \{i_{\text{next}}+1,\ldots,n\}} \mathbf{E}\left[v_i\right]\right\}.\end{aligned}$$

Furthermore, if $i_{\text{next}} = n$ and $k' \geq 1$, then further tests can only be conducted on item n, and they do not harm, so

$$\begin{aligned}\text{DP}(n, D_{n,R}, &D_{i^\star,R'}, k') := \\ &\frac{1}{2} \cdot \text{DP}(n, D_{n,R+(1)}, D_{i^\star,R'}, k'-1) + \frac{1}{2} \cdot \text{DP}(n, D_{n,R+(0)}, D_{i^\star,R'}, k'-1),\end{aligned}$$

where for a tuple $a = (a_1, \ldots, a_k)$, we let $a + (a_{k+1})$ denote the result of appending a_{k+1} to a, i.e., $(a_1, \ldots, a_{k+1})$. This concludes our discussion of the base cases.

In general, when $i_{\text{next}} \neq n$ and $k' \geq 1$, we have to decide whether to perform a test on item i_{next} or to move on to item $i_{\text{next}} + 1$. The expected value of doing that can be computed similarly to the latter case. Therefore

$$\begin{aligned}\text{DP}(i_{\text{next}}, D_{i_{\text{next}},R}, &D_{i^\star,R'}, k') := \\ &\max\Big\{\frac{1}{2} \cdot \text{DP}(i_{\text{next}}, D_{i_{\text{next}},R+(1)}, D_{i^\star,R'}, k'-1) \\ &\qquad + \frac{1}{2} \cdot \text{DP}(i_{\text{next}}, D_{i_{\text{next}},R+(0)}, D_{i^\star,R'}, k'-1), \\ &\qquad \text{DP}(i_{\text{next}}+1, D_{i_{\text{next}}+1,()}, D^\star, k')\Big\},\end{aligned}$$

where $()$ denotes the null tuple, and $D^\star$ is the (in case of a tie, any) distribution of $D_{i_{\text{next}},R}$ and $D_{i^\star,R'}$ that maximizes the expected value drawn from the respective distribution. Note that $D_{i_{\text{next}}+1,\emptyset} = D_{i_{\text{next}}+1}$.

Then $\text{DP}(1, D_{1,()}, \emptyset, k)$ contains the expected value extracted by the optimal testing strategy. To obtain the optimal strategy, we perform the profit-maximizing action at all times (as usual). As a conclusion, Theorem 4.2 follows.

5. Conclusion. A strong and, arguably, unrealistic assumption in existing stochastic probing models is that every probe reveals full information about the probed item. We initiate research that addresses this shortcoming and introduce a first natural model where repeated testing of a single item gradually reveals more information. For this model, we provide polynomial-time algorithms with constant approximation

This manuscript is for review purposes only.

factors for both i.i.d. and general independent, non-negative distributions. We also tightly bound the adaptivity gap to a logarithmic factor.

An interesting direction for future work are hardness results for stochastic probing problems. Only little is known about computational hardness in the standard model of probing: Computing the best *non-adaptive* strategy for a closely related standard probing model is known to be NP-hard [16]. For a large class of such stochastic optimization problems hardness (sometimes even w.r.t. #P) is merely conjectured [14]. In the context of our work, tight lower bounds for the ratio of optimal probing and testing strategies, or the approximability of the optimal testing algorithm are fascinating open problems.

Another direction for future work is to consider correlated random variables. For related problems in online stopping, the versions with correlations are sometimes hopeless [22], and only few positive results are known [24].

More generally, there is potential for extending the rich theory on standard probing models towards tests that yield only limited information, including cases in which the learner can choose a set of items instead of a single one.

Acknowledgements. Martin Hoefer was supported by DFG grants Ho 3831/5-1, 6-1 and 7-1. Kevin Schewior was supported in part by the Independent Research Fund Denmark, Natural Sciences, grant DFF-0135-00018B.

REFERENCES

[1] M. Adamczyk, M. Sviridenko, and J. Ward, *Submodular stochastic probing on matroids*, in Symp. Theoretical Aspects of Computer Science (STACS), 2014, pp. 29–40.

[2] S. Agrawal, J. Sethuraman, and X. Zhang, *On optimal ordering in the optimal stopping problem*, in ACM Conf. Economics and Computation (EC), 2020, pp. 187–188.

[3] S. Alaei, *Bayesian combinatorial auctions: Expanding single buyer mechanisms to many buyers*, SIAM J. Comput., 43 (2014), pp. 930–972.

[4] A. Asadpour and H. Nazerzadeh, *Maximizing stochastic monotone submodular functions*, Manag. Sci., 62 (2016), pp. 2374–2391.

[5] A. Asadpour, H. Nazerzadeh, and A. Saberi, *Stochastic submodular maximization*, in Workshop on Internet and Network Economics (WINE), 2008, pp. 477–489.

[6] N. Bansal, A. Gupta, J. Li, J. Mestre, V. Nagarajan, and A. Rudra, *When LP is the cure for your matching woes: Improved bounds for stochastic matchings*, Algorithmica, 63 (2012), pp. 733–762.

[7] H. Beyhaghi and R. Kleinberg, *Pandora's problem with nonobligatory inspection*, in ACM Conf. Economics and Computation (EC), 2019, pp. 131–132.

[8] A. Bhalgat, A. Goel, and S. Khanna, *Improved approximation results for stochastic knapsack problems*, in ACM-SIAM Symp. Discrete Algorithms (SODA), 2011, pp. 1647–1665.

[9] S. Chawla, J. Hartline, D. Malec, and B. Sivan, *Multi-parameter mechanism design and sequential posted pricing*, in ACM Symp. Theory of Computing (STOC), 2010, pp. 311–320.

[10] N. Chen, N. Immorlica, A. R. Karlin, M. Mahdian, and A. Rudra, *Approximating matches made in heaven*, in Int. Colloq. Automata, Languages and Programming (ICALP), 2009, pp. 266–278.

[11] J. R. Correa, P. Foncea, R. Hoeksma, T. Oosterwijk, and T. Vredeveld, *Recent developments in prophet inequalities*, SIGecom Exch., 17 (2018), pp. 61–70.

[12] B. C. Dean, M. X. Goemans, and J. Vondrák, *Approximating the stochastic knapsack problem: The benefit of adaptivity*, Math. Oper. Res., 33 (2008), pp. 945–964.

[13] W. Feller, *An introduction to probability theory and its applications*, John Wiley & Sons, Inc., 1957.

[14] H. Fu, J. Li, and P. Xu, *A PTAS for a class of stochastic dynamic programs*, in Int. Colloq. Automata, Languages, and Programming (ICALP), vol. 107, 2018, pp. 56:1–56:14.

[15] J. C. Gittins, *Bandit processes and dynamic allocation indices*, J. Royal Stat. Soc. Ser. B, (1979), pp. 148–177.

[16] A. Goel, S. Guha, and K. Munagala, *How to probe for an extreme value*, ACM Trans. Algorithms, 7 (2010), pp. 12:1–12:20.

This manuscript is for review purposes only.

[17] D. Golovin and A. Krause, *Adaptive submodularity: Theory and applications in active learning and stochastic optimization*, J. Artif. Intell. Res., 42 (2011), pp. 427–486.
[18] A. Gupta, H. Jiang, Z. Scully, and S. Singla, *The markovian price of information*, in Integer Programming and Combinatorial Optimization (IPCO), 2019, pp. 233–246.
[19] A. Gupta and V. Nagarajan, *A stochastic probing problem with applications*, in Integer Programming and Combinatorial Optimization (IPCO), 2013, pp. 205–216.
[20] A. Gupta, V. Nagarajan, and S. Singla, *Algorithms and adaptivity gaps for stochastic probing*, in ACM-SIAM Symp. Discrete Algorithms (SODA), 2016, pp. 1731–1747.
[21] A. Gupta, V. Nagarajan, and S. Singla, *Adaptivity gaps for stochastic probing: Submodular and XOS functions*, in ACM-SIAM Symp. Discrete Algorithms (SODA), 2017, pp. 1688–1702.
[22] T. P. Hill and R. P. Kertz, *A survey of prophet inequalities in optimal stopping theory*, Contemp. Math., 125 (1992), pp. 191–207.
[23] M. Hoefer, K. Schewior, and D. Schmand, *Stochastic probing with increasing precision*, in Int. Joint Conf. Artif. Intell. (IJCAI), 2021, pp. 4069–4075.
[24] N. Immorlica, S. Singla, and B. Waggoner, *Prophet inequalities with linear correlations and augmentations*, in ACM Conf. on Economics and Computation (EC), 2020, pp. 159–185.
[25] U. Krengel and L. Sucheston, *Semiamarts and finite values*, Bull. Amer. Math. Soc., 83 (1977), pp. 745–747.
[26] B. Lucier, *An economic view of prophet inequalities*, SIGecom Exch., 16 (2017), pp. 24–47.
[27] W. Ma, *Improvements and generalizations of stochastic knapsack and markovian bandits approximation algorithms*, Math. Oper. Res., 43 (2018), pp. 789–812.
[28] S. Singla, *The price of information in combinatorial optimization*, in ACM-SIAM Symp. Discrete Algorithms (SODA), 2018, pp. 2523–2532.
[29] M. L. Weitzman, *Optimal search for the best alternative*, Econometrica, 47 (1979), pp. 641–654.

This manuscript is for review purposes only.

Appendix A. Proof of Lemma 2.2. The sequence of tests can be seen as a sequence of Benoulli trials. $\mathrm{ALG}_{\mathrm{iid}}$ can run k tests, and we are looking for a success run of length $r = \log_2(k')$ in a sequence of k Benoulli trials. This implies that for some item we have succeeded to verify that it is a good item. To avoid trivialities, we assume $r > 1$. Each trial has a success probability of $1 - c = 1/2$. Feller [13, Volume 1, page 325] observes that the probability of no success run of length r is given by

$$q = A_1 + A_2 + \dots A_r \ ,$$

where

$$A_1 = \frac{1-(1-c)x}{(r+1-rx)\cdot c} \cdot \frac{1}{x^{k+1}} \quad \text{and} \quad |A_i| \le \frac{2(1-c)^{k+2}}{rc(2-c)} \ , \quad \text{for all } i = 2, \dots r.$$

Here, x is the root with smallest absolute value of $f(y) = 1 - y + c(1-c)^r \cdot y^{r+1}$. The unique positive root of $f(y)$ that is different from 2 happens to be the one with smallest absolute value. In Lemma A.1 below we show that with $c = 1/2$, this root satisfies $1 + \frac{1}{2k'} \le x \le 1 + \frac{1}{k'}$. This allows to conclude

$$\begin{aligned}
q &\le A_1 + (r-1)\frac{2}{2^{k+2}\cdot r \cdot \frac{1}{2}\cdot\frac{3}{2}} \\
&= \frac{1-\frac{1}{2}x}{(r+1-rx)\frac{1}{2}} \cdot \frac{1}{x^{k+1}} + \frac{r-1}{r}\cdot\frac{1}{2^{k-1}3} \\
&\le \frac{1-\frac{1}{2k'}}{1-\frac{r}{k'}} \cdot \frac{1}{(1+\frac{1}{2k'})^{k+1}} + \frac{1}{2^{k-1}3} \\
&= \frac{1-\frac{1}{2k'}}{1-\frac{\log_2(k')}{k'}} \cdot \frac{1}{(1+\frac{1}{2k'})^{(2k'+1)\frac{k+1}{2k'+1}}} + \frac{1}{2^{k-1}3} \\
&\le \frac{1-\frac{1}{2k'}}{1-\frac{\log_2(k')}{k'}} \cdot \frac{1}{\mathrm{e}^{\frac{k+1}{2k'+1}}} + \frac{1}{2^{k-1}3} \\
&\le \frac{1-\frac{1}{2k'}}{1-\frac{\log_2(k')}{k'}} \cdot \frac{1}{\sqrt{\mathrm{e}^{\frac{k+1}{k'+1}}}} + \frac{1}{2^{k-1}3} \ ,
\end{aligned}$$

for all $k > 1$. Here, we used $(1+1/x)^{x+1} > \mathrm{e}$ in the second to last inequality.

Lemma A.1. *Let $r \in \mathbb{N}_{\ge 2}$, and x_0 be the unique positive root of $f(y) = 1 - y + \left(\frac{y}{2}\right)^{r+1}$ that is different from 2. Then, $1 + \frac{1}{2^{r+1}} \le x_0 \le 1 + \frac{1}{2^r}$.*

Proof. Feller [13] observes that $f(y)$ has a unique positive root that is different from 2. Obviously, f is continuous. We show that $f(1+\frac{1}{2^{r+1}}) > 0$, and $f(1+\frac{1}{2^r}) < 0$ for all $r \in \mathbb{N}_{\ge 2}$. First, we note that

$$f\left(1+\frac{1}{2^{r+1}}\right) = 1 - \left(1+\frac{1}{2^{r+1}}\right) + \frac{1}{2^{r+1}}\left(1+\frac{1}{2^{r+1}}\right)^{r+1} > -\frac{1}{2^{r+1}} + \frac{1}{2^{r+1}}\cdot 1 = 0 \ .$$

Second, for the case $r \ge 3$ we observe

$$\begin{aligned}
f\left(1+\frac{1}{2^r}\right) &= 1 - \left(1+\frac{1}{2^r}\right) + \frac{1}{2^{r+1}}\left(1+\frac{1}{2^r}\right)^{r+1} \\
&= -\frac{1}{2^r} + \frac{1}{2^{r+1}}\left(1+\frac{1}{2^r}\right)^r\left(1+\frac{1}{2^r}\right)
\end{aligned}$$

This manuscript is for review purposes only.

$$= \frac{1}{2^{r+1}} \left(-2 + \left(1 + \frac{1}{2^r}\right)^r \left(1 + \frac{1}{2^r}\right)\right)$$
$$< \frac{1}{2^{r+1}} \left(-2 + \mathrm{e}^{\frac{r}{2^r}} \left(1 + \frac{1}{2^r}\right)\right) .$$

We note that $\frac{r}{2^r}$ is at most 3/8. Thus,

$$f\left(1 + \frac{1}{2^r}\right) < \frac{1}{2^{r+1}} \left(-2 + \mathrm{e}^{\frac{3}{8}} \left(1 + \frac{1}{2^3}\right)\right) < \frac{1}{2^{r+1}} (-2 + 2) = 0 .$$

For $r = 2$, we can easily check that $f\left(1 + \frac{1}{4}\right) = -3/512 < 0$, which finishes the proof.□

Appendix B. Testing for a c-quantile. As mentioned above, the analysis can be extended rather generically to the case when each test reveals if the realization is above or below a c-quantile of the conditional distribution for an item, for any constant $c \in (0, 1)$. Then, using

$$k' = \left(\frac{1}{1-c}\right)^{\lceil \log_{1/(1-c)} \min\{n,k+1\} \rceil} ,$$

we define a good item as one where the first $r = \log_{1/(1-c)}(k') = \lceil \log_{1/(1-c)}(\min\{n, k+1\}) \rceil$ tests are all positive. The probability that we get such a item can be bounded by generalizing Lemma 2.2 from $c = 1/2$ to $c \in (0, 1)$. In particular, a calculation similar to the one in Lemma A.1 shows that the smallest root of $f(y) = 1-y+c(1-c)^r \cdot y^{r+1}$ is between $1 + (1-c)^r c \leq x_0 \leq 1 + (1-c)^r$ when $r = \omega(1)$ is sufficiently large:

- For $y = 1 + (1-c)^r$

$$\begin{aligned} f(1 + (1-c)^r) &= 1 - (1 + (1-c)^r) + c(1-c)^r(1 + (1-c)^r)^{r+1} \\ &= (1-c)^r(c(1 + (1-c)^r)^{r+1} - 1) < 0 \end{aligned}$$

  holds if and only if $c(1 + (1-c)^r)^{r+1} < 1$, or

  (B.1) $$(r + 1)\ln(1 + (1-c)^r) < \ln 1/c \ .$$

  Since $\ln(1+x) \leq x$ for all $x \geq 0$, a sufficient condition for (B.1) is $(r+1)(1-c)^r < \ln 1/c$. This holds for $r = \omega(1)$ since $(r + 1)(1-c)^r$ is exponentially decreasing in r, while $\ln 1/c$ is a constant.
- For $y = 1 + (1-c)^r c$

$$\begin{aligned} f(1 + (1-c)^r c) &= 1 - (1 + (1-c)^r c) + c(1-c)^r(1 + (1-c)^r c)^{r+1} \\ &= (1-c)^r c((1 + (1-c)^r c)^{r+1} - 1) > 0 \end{aligned}$$

  holds since $(1-c)^r c > 0$ and $(1 + (1-c)^r c)^{r+1} > 1$ whenever $c \in (0, 1)$.

Using these bounds in Lemma 2.2, the probability q to find no good item is again dominated by the factor $\frac{1}{x^{k+1}}$ which is at most

$$\frac{1}{(1 + c(1-c)^r)^{k+1}} = \frac{1}{\left(1 + \frac{c}{k'}\right)^{k+1}} \leq \frac{1}{\left(1 + \frac{c(1-c)}{k+1}\right)^{k+1}} = \frac{1}{\mathrm{e}^{c(1-c)}} + o(1).$$

As such, the probability to find a good item is at least

$$\alpha_c = 1 - \frac{1}{\mathrm{e}^{c(1-c)}} - o(1),$$

which bounds the approximation ratio of the algorithm.

This manuscript is for review purposes only.