Knapsack Secretary with Bursty Adversary

Thomas Kesselheim
University of Bonn, Germany
thomas.kesselheim@uni-bonn.de

Marco Molinaro
PUC-Rio, Rio de Janeiro, Brazil
mmolinaro@inf.puc-rio.br

Abstract

The random-order or *secretary* model is one of the most popular beyond-worst case model for online algorithms. While this model avoids the pessimism of the traditional adversarial model, in practice we cannot expect the input to be presented in perfectly random order. This has motivated research on *best of both worlds* (algorithms with good performance on both purely stochastic and purely adversarial inputs), or even better, on inputs that are a *mix* of both stochastic and adversarial parts. Unfortunately the latter seems much harder to achieve and very few results of this type are known.

Towards advancing our understanding of designing such robust algorithms, we propose a random-order model with *bursts* of adversarial time steps. The assumption of burstiness of unexpected patterns is reasonable in many contexts, since changes (e.g. spike in a demand for a good) are often triggered by a common external event. We then consider the *Knapsack Secretary* problem in this model: there is a knapsack of size k (e.g., available quantity of a good), and in each of the n time steps an item comes with its value and size in $[0,1]$ and the algorithm needs to make an irrevocable decision whether to accept or reject the item.

We design an algorithm that gives an approximation of $1 - \tilde{O}(\Gamma/k)$ when the adversarial time steps can be covered by $\Gamma \geq \sqrt{k}$ intervals of size $\tilde{O}(\frac{n}{k})$. In particular, setting $\Gamma = \sqrt{k}$ gives a $(1 - O(\frac{\ln^2 k}{\sqrt{k}}))$-approximation that is resistant to up to a $\frac{\ln k}{\sqrt{k}}$-fraction of the items being adversarial, which is almost optimal even in the absence of adversarial items. Also, setting $\Gamma = \tilde{\Omega}(k)$ gives a constant approximation that is resistant to up to a constant fraction of items being adversarial. While the algorithm is a simple "primal" one it does not possess the crucial symmetry properties exploited in the traditional analyses. The strategy of our analysis is more robust and significantly different from previous ones, and we hope it can be useful for other beyond-worst-case models.

2012 ACM Subject Classification Theory of computation → Online algorithms

Keywords and phrases Beyond worst-case, secretary problem, random order, online algorithms, knapsack

Digital Object Identifier 10.4230/LIPIcs.ICALP.2020.72

Category Track A: Algorithms, Complexity and Games

Related Version The full version is available at `https://arxiv.org/abs/2006.11607`.

Funding This work was done in part while the authors were visiting the Simons Institute for the Theory of Computing.
Marco Molinaro: MM was supported by CNPq grants Universal #431480/2016-8 and Bolsa de Produtividade em Pesquisa #310516/2017-0, FAPERJ grant Jovem Cientista do Nosso Estado, and a Microsoft Research Fellowship at the Simons Institute for the Theory of Computing.

1 Introduction

In standard competitive analysis of online algorithms, one assumes that an adversary completely defines the input. While this is a useful model for designing algorithms for many problems, for many others this model is too pessimistic and no algorithm can outperform the trivial ones. One classical example is the *Secretary Problem* and its generalizations. In this problem, one is presented a sequence of n items of values $v_1, \ldots, v_n$. Upon each arrival, one has to decide *irrevocably* if one accepts or rejects the item, without knowing the value of future items in the sequence. The goal is to select a single item in order to maximize the value obtained. It is easy to see that in the adversarial model the best guarantee possible is to obtain expected value that is a $\frac{1}{n}$-fraction of the offline optimum, and this is achieved by the trivial algorithm that chooses one of the n time steps at random and blindly accepts the item in this time step.

In order to avoid the pessimism of this model and allow for the design of non-trivial algorithms with *hopefully* better performance in practice, there has been a push to consider *beyond worst-case* models. One of the most prominent such models is the *random-order model*, where the adversary can choose the set of items in the instance, but they are presented in uniformly random order. This model has been studied since at least the 60s and has seen a lot of developments in the past decade, and several problems are now well-understood under this model, such as Knapsack and more generally Packing LPs [21, 3, 2, 28, 18, 16, 1], assignment problems [8, 12, 18], matroid optimization [4, 6, 23, 13, 14], and many more. For example, for the Secretary Problem in the random-order model one can obtain a $\frac{1}{e}$-fraction of the offline optimal value (as $n \to \infty$) with the following classical threshold-based algorithm: reject the first $\frac{1}{e}$-fraction of items but note their maximum value, then select the next element which exceeds this value if such an element appears.

However, in practice we cannot expect the sequence to arrive exactly in random order. This has motivated research on *best of both worlds*, namely algorithms with good performance on both purely stochastic and purely adversarial inputs [25, 26, 22, 27]. Even more interesting are algorithms that work well on inputs that are a *mix* of both stochastic and adversarial parts. But this seems to be much harder to achieve: in online algorithms we are only aware of the results of [11] on budgeted allocation (see Section 1.3 for a description of their model and assumptions), while in online learning results of this type have only been obtained very recently for multi-armed bandits [29, 24, 30, 15]. We note that all these results are for settings in which non-trivial guarantees can be achieved for pure adversarial inputs.

Towards advancing our understanding of designing such robust algorithms, we introduce a model that mixes random-order and adversarial time steps, assuming that the latter comes in *bursts*. The random-order times represent when the environment is in a "stationary" or "predictable" state, while the adversarial times represent "unexpected" patterns. The assumption of burstiness of unexpected patterns is reasonable in many contexts, since changes are often triggered by an external common event, e.g., the surge in gun sales after news of possible changes in gun control regulations. See [19, 20, 9, 7] for examples of the different ways in which burstiness can be modeled and areas of applications.

1.1 The *Bursty Adversary plus Random Order* (BARO) model

We describe more formally the general version of the proposed model BARO. Consider an online problem where decisions are made sequentially and irrevocably at times $1, 2, \ldots, n$. In our model, the adversary first chooses some the time steps $Adv \subseteq [n]$ to be "adversarial" and leaves the others $RO = [n] \setminus Adv$ as "random-order" times. In order to capture the burstiness of the adversarial time steps in a clean way, let $\mathbb{W}$ be the partition of $[n]$ into disjoint intervals of length ℓ. We then assume that the adversarial times Adv are covered by at most Γ intervals in $\mathbb{W}$. Notice that this allows various patterns in the adversarial part of the input, including individual (non-bursty) adversarial times as well as bursts of size much larger than ℓ, for a total of up to $\Gamma\ell$ adversarial times. As in the standard random-order model, the items/inputs on the random-order times RO are arbitrary but presented in uniformly random order. The *sequence* items/inputs on the adversarial times Adv is fully adversarial that can be adaptively generated based on an algorithm's behavior and may even depend on the *order* of the items in RO.

It is important to highlight that the algorithm does not know which time steps are adversarial or random-order, and that in each time step only one item arrives (i.e., the adversarial items do not come in batches).

Note that in many problems this adversary can make an instance completely adversarial by sending "dummy" random-order items. For example, in the Secretary Problem the adversary can set the value of all random-order items to be 0; so again no non-trivial guarantees is possible in this case. In order to obtain meaningful guarantees, we compare the algorithm's performance only to the optimum over the random-order times RO, which we denote by OPT_{RO}. Thus, in a maximization problem we say that an algorithm is *α-competitive* in the BARO model if the expected value of the algorithm is at least $\alpha \mathrm{OPT}_{RO}$.

1.2 Our Results

In this paper we use the BARO model to obtain a more robust algorithm for the *Knapsack Secretary* problem, a well-studied generalization of the Secretary Problem. The offline version of the problem is the standard Knapsack Problem: there are n items, each with a value $v_i \geq 0$ and size $w_i \in [0, 1]$, and we have a knapsack of size k; the goal is to select a subset of items with total size at most k, and with total value as large as possible.

Our main result is an algorithm for the Knapsack Problem in the BARO model that is resistant to a fraction of items being adversarial.

▶ **Theorem 1.** *There is a $\big(1 - O\big(\frac{\Gamma\ell}{n} \ln \frac{n}{\Gamma\ell}\big)\big) = \big(1 - O\big(\frac{\Gamma \ln k}{k} \ln \frac{k}{\Gamma \ln k}\big)\big)$-competitive algorithm for the Knapsack Problem in the BARO model where the adversarial times can be covered by $\Gamma \geq \sqrt{k}$ windows of size $\ell = \frac{n \ln k}{k}$.*

Notice that the term $\frac{\Gamma\ell}{n}$ in the guarantee is precisely the fraction of adversarial items that the algorithm can cope with. For example, setting $\Gamma = \sqrt{k}$, our algorithm obtains a $(1 - O(\frac{\ln^2 k}{\sqrt{k}}))$-approximation in the presence of up to a $O(\frac{\ln k}{\sqrt{k}})$-fraction of items being adversarial. For large k this approximation is almost optimal: even in the absence of adversarial items (and even when all items are unit-sized) the best approximation possible is $1 - \Omega(\frac{1}{\sqrt{k}})$ [21] (and this is achieved for example by [28, 18, 1, 16]). Note that these competitive ratios go to 1 as the budget $k \to \infty$ (recall the normalization of sizes being at most 1). Moreover, with $\Gamma = \Omega(\frac{n}{\ell})$ the algorithm achieves a constant approximation in the presence of a constant fraction of adversarial items.

Primal Algorithm with Time-Based Constraints. Our starting point is the *primal* strategy for the random-order model, whose high-level idea is the following: At time t, one solves a knapsack LP with the items seen so far but with budget proportionally scaled to be $\lceil \frac{t}{n} k \rceil$, and pick (a fraction of) the item at time t exactly as prescribed by the optimal LP solution, if there is space available in the full budget of k.

While this strategy obtains the optimal guarantee in the random-order model [18], it fails in the presence of adversarial items. One way in which it fails is by picking "too many items": Suppose that the first k items are adversarial, have size 1, and they all have infinitesimal values but sorted in increasing order, and the random-order items have all value and size equal to 1; it is easy to see that the primal algorithm will pick all the adversarial items, filling up the budget with items of infinitesimal value. (Similar examples exist where the adversarial items are not in the beginning of the sequence.) To counter this, in our algorithm we add additional restrictions, outside of the LP, that the algorithm can only pick a constant mass of items in each window of size $\ell \approx \frac{n}{k}$, which is roughly the behavior of the optimal solution if the n items were in random order.

However, the algorithm may now fail by picking "too few" items: consider the same example as before but now all the adversarial items have value $1+\varepsilon$, thus slightly more valuable than the random-order items. The algorithm will then only pick 1 of these adversarial items (by the new restriction added) and will not pick any of the random-order items, since the LP will always fill up its budget with the better adversarial items; so the algorithm obtains value $1+\varepsilon$, while the $\mathrm{OPT}_{RO} = k$. To avoid this, we also add additional constraints *to the LP* that its solution can select at most a constant number of items in each window of size $\ell \approx \frac{n}{k}$ (note there are $\frac{t}{n}k$ disjoint such windows in $[t]$ and the LP selects total size $\approx \frac{t}{n}k$, again on average 1 per window).

The main difficulty is analyzing the algorithm in the presence of the additional restrictions/constraints. Previous analyses of primal-style algorithms crucially relied on the fact the LP (and its optimal solution) was invariant to the permutation of items/coordinates. This brings about some crucial independence properties: Decisions at time t are independent of the *order* of the arrivals at times $1, \ldots, t-1$ and therefore of the respective decisions. This property allows for the direct use of known concentration inequalities to control the total occupation incurred by the algorithm.

Since our new restrictions/constraints are not permutation invariant, we need to use a different type of analysis. The main handle is what we call the *weighted rank* of an item: the sum of the weights of items with higher value density $\frac{v_i}{w_i}$ than this item, divided by the knapsack capacity. That is, it is by how much one would have to scale the knapsack capacity before the offline optimum would start picking this item. The very high-level idea of the analysis is intuitive: The higher the weighted rank of an item, the smaller its probability of being picked by the LP, even with the new constraints. In addition, while there are complicated dependencies between the events "the algorithms picks the item at time t", the weighted ranks of the items in the random-order times are almost independent: they are just sampled without replacement. We leverage this to obtain custom concentration inequalities that control the algorithm's occupation of the different restrictions/constraints.

1.3 Related Work

As already pointed out above, many algorithms have been proposed for online optimization problems with random arrival order. However, these algorithms usually break when moving to the BARO model. For concreteness, let us illustrate the effect on Kleinberg's algorithm [21] for the multiple-choice secretary problem, a special case of our problem. The algorithm is allowed up to k selections. Throughout the sequence, it never picks items which are not among the best k so far. Therefore, we can construct the following counterexample. Consider a sequence starting with an adversarial burst of k items of very high value, followed by a random-order sequence with items of smaller values. On this sequence, the algorithm will not pick any random-order items at all. If $n \gg k$, then with high probability (over the randomness of the algorithm) none of the adversarial items are picked either (the threshold-based algorithm for the secretary problem is applied to the first $\approx \frac{n}{k}$ items w.h.p., in which case the first $\approx \frac{1}{e}\frac{n}{k} \gg k$ items are rejected). This argument transfers immediately to other algorithms, such as [2, 18]. Other algorithms such as the one by [1] or by [4] use the beginning of the sequence to estimate the optimal value, which also fails in this sequence.

There is only surprisingly little work when it comes to non-uniform random order model. Recently, [17] introduced models where the order of the items is "much less random" than the uniform random order. Among other results, they show that it is possible to obtain constant-competitive algorithms for the Multiple-choice Secretary Problem under these weaker assumptions, and quantify the minimum entropy of the distribution over orders that admits constant-competitive algorithms for the Secretary Problem. We remark that these models do not explicitly contain adversarial items.

Closer in spirit to our model, [11] consider online budgeted allocation in an online model that mixes both stochastic and adversarial inputs. They provide algorithms that are optimal when the input is totally adversarial, and whose performance improves when the instance becomes "more stochastic". There are two crucial differences between our proposed model and Esfandiari et al.'s model: in the latter, while the adversarial items may appear at any point in the sequence (i.e., no burstiness assumption), it is assumed that the algorithm *knows* the distribution of the items in the non-adversarial times, unlike in our model. Also, unlike the Knapsack Problem studied here, the budgeted allocation problem has constant-competitive algorithms even in the adversarial model. Thus, while to some extent an algorithm does not need to worry about "losing everything" if it is fooled by the adversarial part of the instance, its design and analysis have to be delicate enough to obtain fine control over the constants in the competitive-ratio in order to yield interesting results.

In a very recent paper, Bradac et al. [5] present several results for robust secretary problems in a mixed model very similar to ours, which was inspired by a discussion about a preliminary version of this present paper. In contrast to our model, there is no assumption on the number or burstiness of adversarial rounds, making the results incomparable. Our focus is to understand situations in which we are close to the optimal guarantee without adversarial rounds. Since their adversary is more powerful, the guarantees are worse in two ways: (i) Their benchmark is weakened by leaving out the best item. (ii) The guarantees depend on the overall number of rounds n, whereas ours only depend on k. The techniques are also quite different.

2 BARO Knapsack: model and algorithm

Model. We consider an online knapsack problem. The algorithm knows upfront the knapsack size k and the number of items n, and the items are presented online, one-by-one. In the t-th time step, the current item's value V_t and size W_t are revealed, and the algorithm needs to irrevocably decide what fraction $X_t^{alg} \in [0,1]$ of this item to select. Our algorithm's selection is always integral, i.e., $X_t^{alg} \in \{0,1\}$, but our point of comparison is the best fractional solution. The selections made by the algorithm need to fit the knapsack, namely $\sum_{t\in[n]} W_t X_t^{alg} \le k$ with probability 1, and it tries to maximize the total value of its selections: $\sum_{t\in[n]} V_t X_t^{alg}$. Importantly, the choice in the t-th step has to be made only knowing $V_1, \dots V_t$ and $W_1, \dots, W_t$ (as well as k and n).

The sequences $V_1, \dots, V_n \ge 0$ and $W_1, \dots, W_n \in [0,1]$ are generated by the following *Bursty Adversary plus Random Order* (BARO) model. Let us fix a window size ℓ, and let $\mathbb{W}$ denote the collection of disjoint windows of size ℓ that partitions the time steps $[n]$, that is, $\mathbb{W} = \{\{1, 2, \dots, \ell\}, \{\ell+1, \dots, 2\ell\}, \dots\}$. For concreteness we will use window size $\ell := \frac{n \ln k}{k}$. The adversary first partitions the n times steps into sets *Adv* (adversarial) and *RO* (random-order) with the property that *Adv* can be covered by Γ windows in $\mathbb{W}$; we use $\mathbb{W}^{adv} \subseteq \mathbb{W}$ to denote one such cover, fixed throughout. The adversary also fixes the items for the random-order times, namely the value/size pairs $(v_1, w_1), (v_2, w_2), \dots, (v_{|RO|}, w_{|RO|})$, with $w_i \in [0,1]$ for all i. Moreover, for each random-order time $t \in RO$, nature samples *without replacement* an index I_t from $\{1, 2 \dots, |RO|\}$, i.e., randomly chooses which random-order item will appear at that time. Then, for each time step t the adversary outputs an item with value V_t and size $W_t \in [0,1]$ as follows:

- (Adversarial) If $t \in Adv$, the adversary outputs an item with arbitrary value V_t and size $W_t \in [0,1]$; this may depend on an algorithm's behavior and on the I_t's.
- (Random-order) If $t \in RO$, the adversary outputs the item indexed by I_t, namely that with value $V_t := v_{I_t}$ and size $W_t := w_{I_t}$.

Note that there is a subtle difference between capital and small letters here. By V_t and W_t, we refer to the value and weight of the item *arriving in the t-th step*. By v_i and w_i we refer to the i-th random-order item specified by the adversary *before the random permutation is applied*. Consequently, V_t and W_t are random variables whereas v_i and w_i are not. Furthermore, since the I_t's are sampled without replacement, the items $((V_t, W_t))_{t \in RO}$ in the random-order times are precisely the items $(v_1, w_1), (v_2, w_2), \dots, (v_{|RO|}, w_{|RO|})$ randomly permuted.

Again we highlight that the algorithm does not know which time steps are adversarial and which are random-order, and that the adversarial items do not come in batches. As mentioned before, the benchmark for comparison is the offline optimum for the problem on the random-order items alone, namely $\mathrm{OPT}_{RO} := \max\{\sum_i v_i x_i : \sum_i w_i x_i \le k,\, x \in [0,1]^{|RO|}\}$.

Algorithm. The algorithm we propose is a modification of the primal method of [18] and can be described as follows. Let $\mathbb{W}_t$ be the collection of windows $\mathbb{W}$ truncated to the prefix $[t]$, namely $\{1, \dots, \ell\}, \{\ell+1, \dots, 2\ell\}, \dots, \{\lfloor \frac{t}{\ell} \rfloor \ell + 1, \dots, t\}$. At time t, in order to compute its selection $X_t^{alg} \in \{0,1\}$ of the current item, the algorithm first finds an optimal solution X^t to the following (random) linear program LP_t:

$$
\begin{aligned}
\max & \sum_{t' \le t} V_{t'} X_{t'} & \\
s.t. & \sum_{t' \le t} W_{t'} X_{t'} \le c_t \frac{t}{n} k & \text{(main inner budget)} \\
& \sum_{t' \in B} W_{t'} X_{t'} \le a_1 \frac{\ell}{n} k, \quad \forall B \in \mathbb{W}_t & \text{(inner constraints)} \\
& X \in [0,1]^t, &
\end{aligned}
$$

where we introduce the slight budget scaling $c_t := (1 - \frac{4\Gamma\ell}{t})$, and set the constant $a_1 := 601$. If $X_t^t > 0$, we say that the algorithm *tentatively picks* the item at time t. The algorithm checks if it can permanently pick this item by verifying whether its past selections $X_1^{alg}, \dots, X_{t-1}^{alg}$ satisfy the following constraints:

$$
\begin{aligned}
& \sum_{t' < t} W_{t'} X_{t'} \le k - 1 & \text{(main budget)} \\
& \sum_{t' \in B_{\text{last}}} W_{t'} X_{t'} \le a_4 \frac{\ell}{n} k - 1, & \text{(outer constraint)}
\end{aligned}
$$

where B_{last} denotes the last window in $\mathbb{W}_{t-1}$, and a_4 is a sufficiently large constant (set in Lemma 13). If so, the algorithm fully picks the item, namely it sets $X_t^{alg} = 1$; otherwise we say that it is *blocked* and it does not pick the item at all, setting $X_t^{alg} = 0$.

To get some intuition why the algorithm is reasonable, let us observe how the "offline optimum" OPT_{RO} builds up over time. We can define random variables X_t^* indicating what fraction of the item arriving at time t is packed in OPT_{RO}. Because the permutation is uniformly random, these random variables are identically distributed for all $t \in RO$. More specifically, we have $\mathbb{E}[V_t X_t^*] = \mathrm{OPT}_{RO}/|RO| \approx \mathrm{OPT}_{RO}/n$ and $\mathbb{E}[W_t X_t^*] \le k/|RO| \approx k/n$. So, *in expectation*, slightly scaled versions of the random variables fulfill all constraints stated above. Our algorithm, of course, does not know X_t^* but tries to mimic this process. Particularly, the goal of (inner constraints) and (outer constraint) is to spread out the choices made by the algorithm over time so that the consequences of adversarial bursts are mitigated.

Notice that by construction the solution X^{alg} returned by the algorithm is always feasible, namely $\sum_{t \le n} W_t X_t^{alg} \le k$. Thus, we only need to argue that it obtains enough value.

▶ **Theorem 2** (Total value). *The expected value of the solution X^{alg} returned by the algorithm satisfies*

$$\mathbb{E}\left[\sum_{t \in RO} V_t X_t^{alg}\right] \ge \left(1 - O\left(\frac{\Gamma \ell}{n} \ln \frac{n}{\Gamma \ell}\right)\right) OPT_{RO}.$$

Roadmap of the analysis. In Section 3 we upper bound for each random-order time t the probability that the algorithm tentatively selects that item. Next, we boost this per-time upper bound into concentration inequalities for the volume of the selections made up to a given point, and use it to upper bound the probability that the algorithm is blocked by constraint (main budget) or (outer constraint), in which case it would not be able to make permanent its tentative selection (Section 4). Using this, we lower bound the value obtained by the algorithm in each (free) random-order time step (Section 5), and add over all such time steps to show that the algorithm obtains the desired value (Section 6). Due to space constraints, the proofs of several lemmas are deferred to the full version of the paper.

Without loss of generality we assume that the random-order times are sorted in decreasing order of value density, namely $\frac{v_1}{w_1} \ge \frac{v_2}{w_2} \ge \ldots \ge \frac{v_{|RO|}}{w_{|RO|}}$. Also, we say that an item is **better** than another if it has higher value density. For simplicity, we also assume that no item has value or weight equal to 0 (else automatically exclude/include in the solution), and that the sum of all item sizes is at least the knapsack size k. We also assume that there are no ties in the value densities $\frac{v_i}{w_i}$; this can be accomplished by infinitesimal perturbations to the values, for example. We also assume $\frac{n}{2} \ge k \ge 80$ and that $\frac{\Gamma \ell}{n} \le \frac{1}{2}$, so at most half of the windows can have adversarial items. With overload of notation, we use I_t to denote the actual item (pair (V_t, W_t)) at time t, even when t is an adversarial time.

3 Controlling tentative selections via weighted rank

We use $T_t := \mathbf{1}(X_t^t > 0)$ to denote the indicator of *tentative* selection by the algorithm at time t. Our goal in this section is to argue that the algorithm does not tentatively select too many items. As mentioned before, the main handle for making this formal is the notion of *weighted rank*. The weighted rank of the random-order item i is a $\frac{1}{k}$ scaling of the sum of the weights of random-order items better than it (recall these items are sorted in decreasing order of value density $\frac{v_i}{w_i}$).

▶ **Definition 3** (Weighted rank). *The* weighted rank *of the random-order item i is $r_i := \frac{1}{k} \sum_{i' < i} w_{i'}$ (we also define $r_{|RO|+1} = \frac{1}{k} \sum_i w_i$ for convenience). For a random-order time t, we use $R_t := r_{I_t}$ to denote the total weighted rank of the item I_t at this time.*

As before, one interpretation of the weighted rank r_i is the following: considering the offline problem with only random-order items, r_i is by how much we need to scale the knapsack of size k before the optimal fractional solution wants to pick a strictly positive fraction of item i. Thus, the higher the rank the worse the item is.

The main result of this section says that the worse the item at time t is, the less likely the algorithm is to tentatively pick it (the extra conditioning on items $(I_{t'})_{t' \in S}$ will be technically useful later and may be ignored throughout at a first read).

▶ **Lemma 4** (UB tentative selection). *Consider a random-order time $t \geq 8\ell(\Gamma+1)$, and a set S of random-order times with $|S| \leq \frac{\ln k}{4}$. Then*

$$\Pr\left(T_t = 1 \;\middle|\; R_t,\, (I_{t'})_{t' \in S}\right) \leq \psi(R_t), \qquad \textit{where } \psi(\gamma) = \begin{cases} 1, & \textit{if } \gamma < 1 \\ \frac{2}{k}, & \textit{if } \gamma \in [1, 50] \\ 4ke^{-\frac{\gamma}{20}\ln k}, & \textit{if } \gamma > 50. \end{cases}$$

For the rest of the section we prove this result. At its heart is the following deterministic monotonicity property of the LP: Fix a scenario (so the LP is deterministic); if there *is* a solution for the LP with only items better than I_t that saturates the main budget, then I_t is not included at all in the *optimal* LP solution. This is clear if we did not have the inner constraints: The optimal LP solution is obtained by the greedy procedure, and if we can saturate the budget with only better items the greedy will stop before reaching I_t. While this does not necessarily hold in the presence of general side constraints, we show it still does under the simple inner constraints. The proof is deferred to the full version of the paper.

▶ **Lemma 5.** *Consider a time $t \in [n]$, and fix a scenario $I_1, I_2, \ldots, I_n$. Suppose that there is a feasible solution $\bar{X}$ of LP_t with $\sum_{t' \leq t} W_{t'} \bar{X}_{t'} = c_t \frac{tk}{n}$ and whose support only includes times with items strictly better than I_t (i.e., $\bar{X}_{t'} > 0$ implies that $I_{t'}$ is strictly better than I_t, for all $t' \in [t]$). Then in any optimal solution X^* of LP_t we have $X^*_t = 0$. (Thus, I_t is not tentatively selected by our algorithm.)*

Our next lemma will leverage this result to show that if there are many items in random-order-only windows better than I_t, then the probability of tentatively selecting the latter is small. Before that, we need to introduce the definition of *free time*, the ones we will focus on for most of the analyses.

▶ **Definition 6** ($\textsc{Free}_t$ and RO_t). *A time is* free *if it does not belong to one of the adversarial windows $\mathbb{W}^{adv}$. We use $\textsc{Free}_t$ to denote the collection of free times in $[t]$. Furthermore, $\mathbb{W}^{free}_t$ denotes the windows from $\mathbb{W}_t$ that only contain free times.*

We also use $RO_t := RO \cap [t]$ to denote all the random-order times (free or otherwise) in $[t]$. With slight abuse in notation, we also use RO_t to denote the cardinality *of RO_t.*

The following estimates follow directly from the assumption that there are at most Γ adversarial windows, each of size ℓ.

▶ **Observation 7.** *The following holds: (a) If $t \geq 2\,\Gamma\ell$ then $|\textsc{Free}_t| \geq \frac{t}{2}$; (b) $\frac{1}{RO_n} \leq \frac{1}{|\textsc{Free}_n|} \leq \frac{1}{n}(1 + \frac{2\Gamma\ell}{n})$.*

▶ **Lemma 8.** *Consider a random-order time $t \geq 2(\Gamma\ell + 1)$. For a value $\gamma \geq 0$, let G_γ be the event that the sum of the sizes of the items in the times $\textsc{Free}_t$ that are better than I_t equals $\gamma c_t \frac{tk}{n}$ (i.e., $\sum_{t' \in \textsc{Free}_t : I_{t'} < I_t} W_{t'} = \gamma c_t \frac{tk}{n}$). Then for any set of random-order times $S \subseteq RO$ with $|S| \leq \frac{\ln k}{4}$, we have*

$$\Pr(T_t = 1 \mid G_\gamma, I_t, (I_{t'})_{t' \in S}) \leq \frac{1}{2}\psi(\gamma) = \begin{cases} \frac{1}{k}, & \textit{if } \gamma \in [1, 50] \\ 2ke^{-\frac{\gamma}{20}\ln k}, & \textit{if } \gamma > 50. \end{cases}$$

Proof. Condition on I_t, $(I_{t'})_{t' \in S}$, and on the *set* of items $\{I_{t'}\}_{t' \in \textsc{Free}_{t-1}}$ in the free times in a way that the event G_γ holds; let ω denote this conditioning. If suffices to show the upper bound $\Pr(T_t = 1 \mid \omega) \leq \frac{1}{2}\psi(\gamma)$, and the lemma follows by taking expectation with respect to multiple of these ω's. Also notice that this conditioning does not fix the *relative order* of the items in $\textsc{Free}_{t-1} \setminus S$, thus: ($\star$) The items at times $\textsc{Free}_{t-1} \setminus S$ are in random order even when conditioning on ω.

Let E be the event that there is a feasible solution X for LP_t whose support only has items better than I_t and that saturates the main budget, i.e., $\sum_{t' \le t} W_{t'} X_{t'} = c_t \frac{tk}{n}$. From Lemma 5, whenever E holds I_t is *not* tentatively selected, so it suffices to *lower bound* the probability $\Pr(E \mid \omega)$.

Case 1: $\boldsymbol{\gamma \in [1, 50]}$**.** If for each of the free windows $\mathbb{W}_{t-1}^{\mathit{free}}$ the total size of items better than I_t in the window is at most $a_1 \ell \frac{k}{n}$ (not "too many good items" in any free window), then any (fractional) selection of these items of total size $c_t \frac{tk}{n}$ gives a feasible solution for LP_t saturating the main budget, so E holds; notice that it is possible to select this much size because we are in the case $\gamma \ge 1$. The intuition is that since the total size of these good items is $\gamma c_t \frac{tk}{n} \le 50 \frac{tk}{n}$, each window should have about $\frac{\ell}{t} \cdot 50 \frac{tk}{n} = 50 \ell \frac{k}{n}$ of their size in it, so with high probability no window has more than $a_1 \ell \frac{k}{n}$ of their size (recall $a_1 \gg 50$). More formally, consider a free window $B \in \mathbb{W}_{t-1}^{\mathit{free}}$. Let $Z_{B \setminus S} = \sum_{t' \in B \setminus S} \mathbf{1}(I_{t'} < I_t) \cdot W_{t'}$ be the sum of sizes of items in $B \setminus S$ better than I_t, and let $Z = \sum_{t' \in \textsc{Free}_{t-1} \setminus S} \mathbf{1}(I_{t'} < I_t) \cdot W_{t'}$. Notice that under the conditioning ω, Z is a fixed number satisfying $Z \le \gamma c_t \frac{tk}{n} \le \gamma \frac{tk}{n}$, and that $Z_{B \setminus S}$ is a sum of terms sampled without replacement from the terms in Z (because of observation ($\star$)). Thus, we have

$$\mathbb{E}\left[Z_{B \setminus S} \mid \omega\right] = \frac{|B \setminus S|}{|\textsc{Free}_{t-1} \setminus S|} \mathbb{E}[Z \mid \omega] \le \frac{\ell}{|\textsc{Free}_{t-1}| - \ln k} \cdot \frac{\gamma t k}{n}$$
$$= \frac{t}{|\textsc{Free}_{t-1}| - \ln k} \cdot \gamma \ell \frac{k}{n} \le 3 \gamma \ell \frac{k}{n},$$

where the last inequality uses the fact that $t \ge 2\Gamma\ell + 1$, Observation 7, and the assumptions $\Gamma \ge \sqrt{k}$ and $k \ge 80$. Moreover, we can apply the Bernstein's Inequality for sampling without replacement (Corollary 2.3 of [16]) conditionally to the sum $Z_{B \setminus S}$ to obtain

$$\Pr\left[Z_{B \setminus S} \ge 600 \ell \frac{k}{n} \;\middle|\; \omega\right] \le 2 \exp\left(-\frac{9}{7} 3 \gamma \ell \frac{k}{n}\right) \le 2 \frac{1}{k^3} \le \frac{1}{k^2},$$

where in the first inequality we also used that $\tau \ge 3 \cdot 3 \gamma \ell \frac{k}{n}$ because $\gamma \le 50$, and in the last inequality that $k \ge 80$. Since $|S| \le \frac{\ln k}{4}$ and each item has size at most 1, the items in $B \cap S$ have total size less than $\ell \frac{k}{n}$. Thus, the conditional probability is at most $\frac{1}{k^2}$ that the total size of items in B better than I_t is at least $a_1 \ell \frac{k}{n}$ ("too many good items"). Since there are fewer than k windows, by taking a union bound over all free windows $B \in \mathbb{W}_t^{\mathit{free}}$ we see that with probability at least $1 - \frac{1}{k}$ none of these windows has too many good items. Thus, $\Pr(E \mid \omega) \ge 1 - \frac{1}{k}$.

Case 2: $\boldsymbol{\gamma > 50}$**.** The number of windows in $\mathbb{W}_{t-1}^{\mathit{free}}$ of size ℓ (i.e., possibly excluding the last window) is at least $\mathit{num} := \frac{t}{\ell} - \Gamma - 1$. If in each such window the total size of items better than I_t is at least $2\ell \frac{k}{n}$ ("good items everywhere"), then one can (fractionally) select up to $2\ell \frac{k}{n}$-mass of them in each window and get a feasible solution for LP_t that saturates the main budget; this saturation is possible because this can give a total of size $(2\ell \frac{k}{n}) \cdot \mathit{num} \ge c_t \frac{t}{n} k$ of these better items, where the last inequality uses $t \ge 2\ell(\Gamma + 1)$. Since in this case event E holds, it suffices to lower bound the probability of having good items everywhere. The intuition again is that by assumption there is total mass $\gamma c_t \frac{tk}{n} \ge 12 \frac{tk}{n}$ of these better items, so each window should have about $\frac{\ell}{t} \cdot 12 \frac{tk}{n} = 12 \ell \frac{k}{n}$ size in it, and with high probability all of them should have at least $2\ell \frac{k}{n}$ size in it.

More precisely, considering any fixed window $B \in \mathbb{W}_{t-1}^{free}$ it is easy to obtain the lower estimate $\mathbb{E}\left[Z_{B\setminus S} \,\middle|\, \omega\right] \geq \frac{\gamma \ell k}{3n}$ and again applying Bernstein's Inequality we get $\Pr\left[Z_{B\setminus S} \leq 2\ell\frac{k}{n} \,\middle|\, \omega\right] \leq 2\exp\left(-\frac{\gamma}{20}\ln k\right)$. Taking a union bound over the at most k such windows, the probability that we have enough good items in each window in $\mathbb{W}_{t-1}^{free}$ of size ℓ is at least $1 - 2ke^{-\frac{\gamma}{20}\ln k}$. This concludes the proof. ◀

To conclude the proof of Lemma 4, we show that the item at time t having rank R_t implies with high probability that G_{R_t} holds (actually that the weight in $\textsc{Free}_t$ of items better than I_t is *at least* $R_t c_t \frac{tk}{n}$); this is a consequence of the definition of rank and concentration. With these final details, presented in the full version of the paper, we have the proof of Lemma 4.

4 Controlling the probability of being blocked

In this section we show that with good probability, when the algorithm tentatively selects an item, it also permanently selects it, i.e., it is not blocked by the constraints (main budget) and (outer constraint). More precisely, let $O_t := W_t X_t^{alg}$ be the actual occupation incurred by the the algorithm at time t. We use F_t to denote the indicator of the event that the algorithm is *not* blocked at time t, i.e., $F_t = 1$ if (again B_{last} is the last window in $\mathbb{W}_{t-1}$)

$$\sum_{t' \in B_{\text{last}}} O_{t'} \leq a_4 \frac{\ell}{n} k - 1 \qquad \text{and} \qquad \sum_{t' < t} O_{t'} \leq k - 1, \tag{4.1}$$

otherwise $F_t = 0$. The following is the main result of this section.

▶ **Lemma 9** (Probability of being blocked). *For all free times $t \geq 8\ell(\Gamma + 2)$, the probability of being blocked is upper bounded as $\Pr(F_t = 0 \mid I_t) \leq \frac{O(1)}{k\left(1 - \frac{t}{n} - a_5 \frac{\Gamma \ln k}{k}\right)^2}$, for some constant a_5.*

To prove this lemma, we will upper bound the probability that either of the two parts of (4.1) is violated. For the first part, this will be Lemma 13. The bound for the second part is Lemma 14. Lemma 9 then follows by a union bound.

While the first part of (4.1) only concerns the occupation from free time steps, the second part also includes non-free ones. To control this second part, we will nonetheless focus on the occupation over the free windows; for non-free windows B the outer constraints guarantee $\sum_{t' \in B} O_{t'} \leq O(\frac{\ell k}{n}) = O(\ln k)$, and so all the Γ of these windows combined can consume only $O(\Gamma \ln k)$ of the budget (so, for example, in the important case $\Gamma = \sqrt{k}$ this is negligible).

For the free time steps, it suffices to upper bound the (permanent) occupation O_t by the *tentative occupation* $O'_t := W_t T_t$: For the algorithm to select the item at time t, it is necessary but not sufficient that $T_t = 1$. Therefore, we have $O_t \leq O'_t$ and we focus on controlling the O'_t's from now on.

As a start, we use Lemma 4 to show that in each free time step the expected tentative occupation $\mathbb{E}[O'_t]$ is at most $\approx \frac{k}{n}$; thus, essentially both (4.1) hold in expectation. While what we actually need is a generalization of this result, we present it to illustrate the techniques in a clearer way.

▶ **Lemma 10** (UB tentative occupation). *For all free times $t \geq 8\ell(\Gamma + 1)$, we have $\mathbb{E}\left[O'_t\right] \leq \frac{k}{RO_n}\left(1 + O\left(\frac{1}{k}\right)\right)$.*

Proof. Since fixing I_t fixes W_t, using Lemma 4 we have

$$\begin{aligned} \mathbb{E}\, O'_t = \mathbb{E}\, W_t T_t &= \mathbb{E}_{I_t}\left[W_t \cdot \mathbb{E}[T_t \mid I_t]\right] \\ &= \mathbb{E}_{I_t}\left[W_t \cdot \Pr(T_t = 1 \mid I_t)\right] \overset{L.4}{\leq} \mathbb{E}_{I_t}\left[W_t \cdot \psi(R_t)\right] = \frac{1}{RO_n} \sum_i w_i\, \psi(r_i). \end{aligned}$$

Since by definition of rank $r_j = \frac{1}{k}\sum_{j'<j} w_{j'}$, we have $r_{i+1} - r_i = \frac{w_i}{k}$, and thus $w_i = k \cdot \int_{r_i}^{r_{i+1}} 1 \, \mathrm{d}x$. Applying this to the last displayed inequality we get

$$\mathbb{E}\, O'_t \leq \frac{k}{RO_n} \sum_i \int_{r_i}^{r_{i+1}} \psi(r_i)\, \mathrm{d}x. \tag{4.2}$$

Since the item sizes are at most 1, we have $r_{i+1} \leq r_i + \frac{1}{k}$ and so $x - \frac{1}{k} \leq r_i$ for all $x \in [r_i, r_{i+1}]$. Thus, as the function ψ is nonincreasing, the right-hand side of (4.2) is at most

$$\frac{k}{RO_n} \sum_i \int_{r_i}^{r_{i+1}} \psi(x - 1/k)\, \mathrm{d}x \leq \frac{k}{RO_n} \int_0^{\infty} \psi(x - 1/k)\, \mathrm{d}x.$$

Finally, inspecting $\psi(x)$ we see that it takes value 1 for $x < 1$, takes value $\frac{2}{k}$ for $x \in [1, 50]$, and has exponential decay $\leq \frac{e^{-x}}{k}$ after that. Thus, it is easy to see that the integral on the right-hand side is at most $1 + O(\frac{1}{k})$. This concludes the proof. ◀

However, what we actually need is to show that (4.1) (with O'_t's) holds with good probability; for that we need concentration inequalities for the sums of the tentative occupations O'_t's. The biggest problem is that the tentative selections induced by the LP are correlated in a non-trivial way. In particular, it is not clear whether they are negatively associated: for example, if the items up to time $t-1$ are all "very good" the algorithm will not tentatively select at times t, $t+1$, etc., indicating possibility of positive correlations on these times. Thus, the O'_t's are also correlated and it is not clear how to apply standard concentrations inequalities.

4.1 Concentration I: controlling the outer constraint

However, as the example above illustrates, we still have hopes of obtaining good *upper bounds* on the probability of multiple tentative selections. In fact, the probability of multiple selection of items $I_{t_1}, \ldots, I_{t_m}$ is at most the probability that the "worst" of these items is selected; more precisely:

▶ **Lemma 11.** *Consider $m \leq \frac{\ln k}{4}$ random-order times $t_1, \ldots, t_m \geq 8\ell(\Gamma + 1)$. Then*

$$\Pr\left(T_{t_1} = \ldots = T_{t_m} = 1 \;\middle|\; R_{t_1}, \ldots, R_{t_m}\right) \leq \psi\left(\max_i R_{t_i}\right).$$

Proof. The inequality follows from the fact $\Pr(X_1 = \ldots = X_m = 1 \mid E) \leq \min_i \Pr(X_i = 1 \mid E)$, Lemma 4, and $\min_i \psi(R_{t_i}) = \psi(\max_i R_{t_i})$ (by the monotonicity of ψ). ◀

The main advantage of this bound is that the ranks R_{t_i} are "almost" independent (they would be independent if the input sequence was generated by sampling items *with* replacement).

Moreover, this lemma allows us to upper bound products of tentative occupation $\prod_i O'_{t_i}$: for this product to be strictly positive, all these items have to be tentatively selected. In fact, one can prove such upper bound using a similar strategy as in Lemma 10, with a main new element: a simple but general comparison for the expectation of a *non-negative* function under sampling with and without replacement, that allow us to work with a decoupled (independent) version $R'_{t_1}, \ldots, R'_{t_m}$ of the ranks. Formally we have the following, which is deferred to the full version of the paper.

▶ **Lemma 12** (Control of products). *Fix a random-order time t. Consider a set of $m \leq \frac{\ln k}{4}$ distinct RO times $t_1, \ldots, t_m$, all of which are at least $8\ell(\Gamma+1)$ and less than t. Then there are constants $a_2, a_3 > 1$ such that* $\mathbb{E}\left[\prod_{i\in[m]} O'_{t_i} \,\middle|\, I_t\right] \leq \left(1+\frac{a_2^m}{k}\right)\left(1+\frac{4m^2}{RO_n}\right)\left(\frac{k}{RO_n}\right)^m \leq \left(a_3\frac{k}{n}\right)^m$. *In particular, choosing $a_2 = 500$ and $a_3 = 8a_2$ is sufficient.*

Finally, these product estimates can be converted into raw moments/tail inequalities using reasonably standard estimates (e.g., Section 3.4 of [10]), giving the desired control of the outer constraint's occupation (the proof is deferred to the full version of the paper).

▶ **Lemma 13** (Control of outer constraints). *Consider a free time $t \geq 8\ell(\Gamma+1)$, and let B be the last window in $\mathbb{W}_{t-1}$. Then* $\Pr\left(\sum_{t'\in B} O'_{t'} > a_4\ell\frac{k}{n} \,\middle|\, I_t\right) \leq \frac{1}{k}$, *where $a_4 \geq 2e^6 a_3$, and a_3 is the constant from Lemma 12.*

4.2 Concentration II: control of main budget

In order to obtain Lemma 9 we need to show that the second part of (4.1) holds with reasonable probability *even when* $t \approx n$; but since $\mathbb{E}O'_t \approx \frac{k}{n}$, the expected cumulative occupation by the end of the game $\mathbb{E}[\sum_{t'=1}^{t} O'_{t'}]$ is $\approx k$ for $t \approx n$, so we do not have much room. So unlike the previous section, we are interested in "'medium deviations", where the variance is the right quantity to look at. While Lemma 12 directly gives that the cumulative variance until time t is $\lesssim (\frac{tk}{n})^2$, we actually need an upper bound of order $O(\frac{tk}{n})$, which is what one would expect from independent Bernoulli's with success probability $\frac{k}{n}$. Since $\mathrm{Var}(Z) = \mathbb{E}Z^2 - (\mathbb{E}Z)^2$, to obtain variance upper bounds we will obtain an upper bound on the second raw moment and a *lower bound on the expectation.*

For that we actually prove concentration for a modified occupation that upper bounds O'_t in every scenario: $\bar{O}'_t = W_t\bar{T}_t$, where $\bar{T}_t := \max\{T_t, \mathbf{1}[R_t \leq 1]\}$, that is $\bar{T}_t$ equals 1 if either $T_t = 1$ or the weighted rank R_t is at most 1. While Lemma 12 still holds for these variables, we can now get almost matching lower bounds on $(\mathbb{E}\sum_{t'=1}^{t}\bar{O}'_{t'})^2$, giving the desired variance bound $\mathrm{Var}[\sum_{t'=1}^{t}\bar{O}'_{t'}] \leq O(\frac{tk}{n})$. Essentially using the bound in expectation from Lemma 10 and Chebychev's inequality with this variance control gives the following (see the full version of the paper).

▶ **Lemma 14** (Control of main budget). *For every random-order time t, the probability we are blocked by the main budget can be upper bounded as* $\Pr\left[\sum_{t'<t} O'_t > k-1 \,\middle|\, I_t\right] \leq \frac{O(1)}{k\left(1-\frac{t}{n}-O\left(\frac{\Gamma\ln k}{k}\right)\right)^2}$.

Taking a union bound over Lemma 13 and Lemma 14 proves Lemma 9.

5 Lower bounding the value obtained

Recall that $X_t^{alg} = T_tF_t$, i.e., the item is permanently selected exactly when it is tentatively selected and it fits the budgets, and that V_t is the value of the item at time t. The following is then our main lower bound on the value obtained by the algorithm.

▶ **Lemma 15** (Value lower bound). *Consider a free time $t \geq 1{,}212\Gamma\ell$. Then* $\mathbb{E}[V_tT_tF_t] \geq \left(c_t - \varepsilon_t - p_t - \frac{2}{k}\right)\frac{OPT_{RO}}{RO_n}$, *where p_t is the bound from Lemma 9 and* $\varepsilon_t = (a_1+3)\frac{\Gamma\ell}{t} + \sqrt{10\ln k}\sqrt{\frac{2n}{tk}}$.

To prove this, the first step is to obtain the following "dual" to Lemma 4, which says that if the item has low (i.e. good) rank then it is fully tentatively selected with good probability.

▶ **Lemma 16.** *For any free time $t \geq 1{,}212\Gamma\ell$ we have that the probability of fully tentatively selecting item I_t satisfies:* $\Pr\left(X_t^t = 1 \;\middle|\; R_t \leq c_t - \varepsilon_t\right) \geq 1 - \frac{1}{k}$.

This is proved via a "dual" of Lemma 5 plus concentration. Given that good items are tentatively selected with good probability and it is likely to fit the budget (Lemma 9), final difficulty in proving Lemma 15 is the correlation between these quantities and the value V_t. This is why most previous bounds actually work even conditioned on the item I_t; notice that fixing I_t, fixes V_t as a deterministic constant. We sketch the argument.

Proof sketch of Lemma 15. Using the non-negativity of V_t, T_t, and F_t, we have

$$\begin{aligned}\mathbb{E}[V_t T_t F_t] = \mathbb{E}_{I_t}\left[V_t \,\mathbb{E}\left[T_t F_t \mid I_t\right]\right] &\geq \mathbb{E}_{I_t}\left(V_t \,\mathbb{E}\left[T_t F_t \mid I_t\right] \;\middle|\; R_t \leq c_t - \varepsilon_t\right) \Pr(R_t \leq c_t - \varepsilon_t)\\ &\geq \left(1 - p_t - \frac{1}{k}\right) \mathbb{E}_{I_t}\left(V_t \;\middle|\; R_t \leq c_t - \varepsilon_t\right) \Pr(R_t \leq c_t - \varepsilon_t),\end{aligned}$$

where the second inequality uses Lemmas 16 and 9 (via the definition of p_t). Since $c_t - \varepsilon_t \approx 1$ and essentially OPT_{RO} gets value exactly from items of rank at most 1, the last two factors in the inequality together give roughly the expected value OPT_{RO} gets in a time step. ◀

6 Wrapping up: finishing the proof of Theorem 2

To finish the proof we just need to add Lemma 15 over all free time steps except the ones very early or very late in the sequence. More precisely, let $t_0 = 1{,}212\Gamma\ell$ and $\gamma = 1 - (a_5 + 1)\frac{\Gamma\ell}{n}$, and define $T = \{t \in \textsc{Free}_n : t_0 \leq t \leq \gamma n\}$. For $t \notin T$, we use the trivial bound $V_t \geq 0$. For the other time steps we use Lemma 15. With the fact $RO_n \leq n$ we have

$$\mathbb{E}\left[\sum_{t \in RO} V_t X_t^{alg}\right] \geq \sum_{t \in T} \mathbb{E}[V_t T_t F_t] \geq \left(\sum_{t \in T} c_t - \sum_{t \in T} \varepsilon_t - \sum_{t \in T} p_t - \frac{2n}{k}\right) \frac{\mathrm{OPT}_{RO}}{n}. \tag{6.3}$$

We bound each of the remaining sums:

- $\sum_{t \in T} c_t = |T| - \sum_{t \in T} \frac{4\Gamma\ell}{t} \geq (n - O(\Gamma\ell)) - \int_{t_0 - 1}^{n} \frac{4\Gamma\ell}{t}\, \mathrm{d}t = n - O(\Gamma\ell \ln \frac{n}{\Gamma\ell})$.
- $\sum_{t \in T} p_t \leq O\left(\frac{1}{k}\right) \cdot \int_0^{\gamma n} \frac{1}{\left(1 - \frac{t}{n} - a_5 \frac{\Gamma \ln k}{k}\right)^2}\, \mathrm{d}t = O\left(\frac{n}{k}\right) \int_0^{\gamma} \frac{1}{(a-x)^2}\, \mathrm{d}x$, where in the last step we set $a = 1 - a_5 \frac{\Gamma \ln k}{k}$ and use change of variables $x = \frac{t}{n}$. The remaining integral equals $\frac{1}{a-x}\Big|_0^{\gamma} \leq \frac{1}{a-\gamma}$. By our setting of γ we have $a - \gamma = \frac{\Gamma\ell}{n}$, so we obtain $\sum_{t \in T} p_t \leq O\left(\frac{n}{\Gamma}\right)$.
- $\sum_{t \in T} \varepsilon_t \leq \int_{t_0 - 1}^{n} (a_1 + 3) \frac{\Gamma\ell}{t}\, \mathrm{d}t + \int_{t_0 - 1}^{n} \sqrt{\frac{20 n \ln k}{kt}}\, \mathrm{d}t \leq O(\Gamma\ell \ln \frac{n}{\Gamma\ell}) + O(\frac{n\sqrt{\ln k}}{\sqrt{k}})$.

Using these bounds on (6.3) and the assumption $\Gamma \geq \sqrt{k}$ concludes the proof of the theorem.

7 Conclusions

A natural follow-up question is how our results could generalize to other settings. In particular, it would be interesting to extend our algorithm and analysis to packing LPs. The difficulty in using our technique is that there is no natural notion similar to the weighted rank for this setting.

It would also be interesting to better understand the limitations and trade-offs in this and similar models. For example, what regimes of parameter allow constant-competitive or $(1 - \varepsilon)$-competitive algorithms?

References

1 Shipra Agrawal and Nikhil R. Devanur. Fast algorithms for online stochastic convex programming. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 1405–1424, 2015. doi:10.1137/1.9781611973730.93.

2 Shipra Agrawal, Zizhuo Wang, and Yinyu Ye. A dynamic near-optimal algorithm for online linear programming. *Operations Research*, 62(4):876–890, 2014. doi:10.1287/opre.2014.1289.

3 Moshe Babaioff, Nicole Immorlica, David Kempe, and Robert Kleinberg. A knapsack secretary problem with applications. In *APPROX-RANDOM*, 2007.

4 Moshe Babaioff, Nicole Immorlica, and Robert Kleinberg. Matroids, secretary problems, and online mechanisms. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07, pages 434–443, 2007.

5 Domagoj Bradac, Anupam Gupta, Sahil Singla, and Goran Zuzic. Robust algorithms for the secretary problem. In Thomas Vidick, editor, *11th Innovations in Theoretical Computer Science Conference, ITCS 2020, January 12-14, 2020, Seattle, Washington, USA*, volume 151 of *LIPIcs*, pages 32:1–32:26. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.ITCS.2020.32.

6 Sourav Chakraborty and Oded Lachish. Improved competitive ratio for the matroid secretary problem. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1702–1712, 2012. doi:10.1137/1.9781611973099.135.

7 G.R. Dattatreya. *Performance Analysis of Queuing and Computer Networks (Chapman & Hall/Crc Computer & Information Science Series)*. Chapman & Hall/CRC, 2008.

8 Nikhil R. Devanur and Thomas P. Hayes. The adwords problem: online keyword matching with budgeted bidders under random permutations. In *EC*, 2009.

9 Qiming Diao, Jing Jiang, Feida Zhu, and Ee-Peng Lim. Finding bursty topics from microblogs. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, ACL '12, pages 536–544, 2012.

10 Devdatt Dubhashi and Alessandro Panconesi. *Concentration of Measure for the Analysis of Randomized Algorithms*. Cambridge University Press, New York, NY, USA, 1st edition, 2009.

11 Hossein Esfandiari, Nitish Korula, and Vahab Mirrokni. Online allocation with traffic spikes: Mixing adversarial and stochastic models. In *Proceedings of the Sixteenth ACM Conference on Economics and Computation*, EC '15, pages 169–186, New York, NY, USA, 2015. ACM. doi:10.1145/2764468.2764536.

12 Jon Feldman, Monika Henzinger, Nitish Korula, Vahab S. Mirrokni, and Clifford Stein. Online stochastic packing applied to display ad allocation. In *ESA*, 2010.

13 Moran Feldman, Ola Svensson, and Rico Zenklusen. A simple o(log log(rank))-competitive algorithm for the matroid secretary problem. In *Proceedings of the Twenty-sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '15, pages 1189–1201, 2015.

14 Moran Feldman, Ola Svensson, and Rico Zenklusen. A framework for the secretary problem on the intersection of matroids. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '18, pages 735–752, 2018.

15 Anupam Gupta, Tomer Koren, and Kunal Talwar. Better algorithms for stochastic bandits with adversarial corruptions. In *Conference on Learning Theory, COLT 2019, 25-28 June 2019, Phoenix, AZ, USA*, pages 1562–1578, 2019. URL: http://proceedings.mlr.press/v99/gupta19a.html.

16 Anupam Gupta and Marco Molinaro. How the experts algorithm can help solve lps online. *Mathematics of Operations Research*, 41(4):1404–1431, 2016. doi:10.1287/moor.2016.0782.

17 Thomas Kesselheim, Robert D. Kleinberg, and Rad Niazadeh. Secretary problems with non-uniform arrival order. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 879–888, 2015. doi:10.1145/2746539.2746602.

18 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 46th Annual ACM Symposium on Theory of Computing*, STOC '14, pages 303–312, New York, NY, USA, 2014. ACM. doi:10.1145/2591796.2591810.

19 Jon Kleinberg. Bursty and hierarchical structure in streams. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '02, pages 91–101, New York, NY, USA, 2002. ACM. doi:10.1145/775047.775061.

20 Jon Kleinberg, Yuval Rabani, and Éva Tardos. Allocating bandwidth for bursty connections. In *Proceedings of the Twenty-ninth Annual ACM Symposium on Theory of Computing*, STOC '97, pages 664–673, New York, NY, USA, 1997. ACM. doi:10.1145/258533.258661.

21 Robert Kleinberg. A multiple-choice secretary algorithm with applications to online auctions. In *SODA*, 2005.

22 Nitish Korula, Vahab Mirrokni, and Morteza Zadimoghaddam. Online submodular welfare maximization: Greedy beats 1/2 in random order. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing*, STOC '15, pages 889–898, 2015. doi:10.1145/2746539.2746626.

23 O. Lachish. O(log log rank) competitive ratio for the matroid secretary problem. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, pages 326–335, October 2014. doi:10.1109/FOCS.2014.42.

24 Thodoris Lykouris, Vahab Mirrokni, and Renato Paes Leme. Stochastic bandits robust to adversarial corruptions. In *STOC 2018*, 2018.

25 A. Meyerson. Online facility location. In *Proceedings of the 42Nd IEEE Symposium on Foundations of Computer Science*, FOCS '01, pages 426–431, Washington, DC, USA, 2001. IEEE Computer Society. URL: http://dl.acm.org/citation.cfm?id=874063.875567.

26 Vahab S. Mirrokni, Shayan Oveis Gharan, and Morteza Zadimoghaddam. Simultaneous approximations for adversarial and stochastic online budgeted allocation. In *Proceedings of the Twenty-third Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '12, pages 1690–1701, 2012.

27 Marco Molinaro. Online and random-order load balancing simultaneously. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '17, pages 1638–1650, 2017.

28 Marco Molinaro and R. Ravi. Geometry of online packing linear programs. In Artur Czumaj, Kurt Mehlhorn, Andrew Pitts, and Roger Wattenhofer, editors, *Automata, Languages, and Programming*, pages 701–713, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

29 Yevgeny Seldin and Aleksandrs Slivkins. One practical algorithm for both stochastic and adversarial bandits. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, pages 1287–1295, 2014. URL: http://proceedings.mlr.press/v32/seldinb14.html.

30 Julian Zimmert and Yevgeny Seldin. An optimal algorithm for stochastic and adversarial bandits. In *The 22nd International Conference on Artificial Intelligence and Statistics, AISTATS 2019, 16-18 April 2019, Naha, Okinawa, Japan*, pages 467–475, 2019. URL: http://proceedings.mlr.press/v89/zimmert19a.html.