Computer Science and Artificial Intelligence Laboratory
Technical Report

MIT-CSAIL-TR-2010-002 February 1, 2010

Submodular Secretary Problem and Extensions

MohammadHossein Bateni, MohammadTaghi Hajiaghayi, and Morteza Zadimoghaddam

massachusetts institute of technology, cambridge, ma 02139 usa — www.csail.mit.edu

Submodular Secretary Problem and Extensions

MohammadHossein Bateni[*] MohammadTaghi Hajiaghayi[†]

Morteza Zadimoghaddam[‡]

Abstract

Online auction is an essence of many modern markets, particularly networked markets, in which information about goods, agents, and outcomes is revealed over a period of time, and the agents must make irrevocable decisions without knowing future information. Optimal stopping theory, especially the classic *secretary problem*, is a powerful tool for analyzing such online scenarios which generally require optimizing an objective function over the input. The secretary problem and its generalization the *multiple-choice secretary problem* were under a thorough study in the literature. In this paper, we consider a very general setting of the latter problem called the *submodular secretary problem*, in which the goal is to select k secretaries so as to maximize the expectation of a (not necessarily monotone) submodular function which defines efficiency of the selected secretarial group based on their overlapping skills. We present the first constant-competitive algorithm for this case. In a more general setting in which selected secretaries should form an independent (feasible) set in each of l given matroids as well, we obtain an $O(l \log^2 r)$-competitive algorithm generalizing several previous results, where r is the maximum rank of the matroids. Another generalization is to consider l knapsack constraints instead of the matroid constraints, for which we present an $O(l)$-competitive algorithm. In a sharp contrast, we show for a more general setting of *subadditive secretary problem*, there is no $\tilde{o}(\sqrt{n})$-competitive algorithm and thus submodular functions are the most general functions to consider for constant competitiveness in our setting. We complement this result by giving a matching $O(\sqrt{n})$-competitive algorithm for the subadditive case. At the end, we consider some special cases of our general setting as well.

[*]`mbateni@cs.princeton.edu`, Princeton University, Princeton, NJ, USA. Part of the work was done while the author was a summer intern in TTI, Chicago, IL, USA. He was supported by NSF ITR grants CCF-0205594, CCF-0426582 and NSF CCF 0832797, NSF CAREER award CCF-0237113, MSPA-MCS award 0528414, NSF expeditions award 0832797, and a Gordon Wu fellowship.

[†]`hajiagha@research.att.com`, AT&T Labs – Research, Florham Park, NJ, USA.

[‡]`morteza@mit.edu`, MIT, Cambridge, MA, USA; Part of the work was done while the author was visiting EPFL, Lausanne, Switzerland.

1 Introduction

Online auction is an essence of many modern markets, particularly networked markets, in which information about goods, agents, and outcomes is revealed over a period of time, and the agents must make irrevocable decisions without knowing future information. Optimal stopping theory is a powerful tool for analyzing such scenarios which generally require optimizing an objective function over the space of stopping rules for an allocation process under uncertainty. Combining optimal stopping theory with game theory allows us to model the actions of rational agents applying competing stopping rules in an online market. This first has been considered by Hajiaghayi et al. [21] which initiated several follow-up papers (see e.g. [4, 5, 6, 22, 26, 30]).

Perhaps the most classic problem of stopping theory is the well-known *secretary problem*. Imagine that you manage a company, and you want to hire a secretary from a pool of n applicants. You are very keen on hiring only the best and brightest. Unfortunately, you cannot tell how good a secretary is until you interview him, and you must make an irrevocable decision whether or not to make an offer at the time of the interview. The problem is to design a strategy which maximizes the probability of hiring the most qualified secretary. It is well-known since 1963 [10] that the optimal policy is to interview the first $t-1$ applicants, then hire the next one whose quality exceeds that of the first $t-1$ applicants, where t is defined by $\sum_{j=t+1}^{n} \frac{1}{j-1} \leq 1 < \sum_{j=t}^{n} \frac{1}{j-1}$; as $n \to \infty$, the probability of hiring the best applicant approaches $1/e$, as does the ratio t/n. Note that a solution to the secretary problem immediately yields an algorithm for a slightly different objective function optimizing the expected value of the chosen element. Subsequent papers have extended the problem by varying the objective function, varying the information available to the decision-maker, and so on, see e.g., [2, 19, 37, 39].

An important generalization of the secretary problem with several applications (see e.g., a survey by Babaioff et al. [5]) is called the *multiple-choice secretary problem* in which the interviewer is allowed to hire up to $k \geq 1$ applicants in order to maximize performance of the secretarial group based on their overlapping skills (or the joint utility of selected items in a more general setting). More formally, assuming applicants of a set $S = \{a_1, a_2, \cdots, a_n\}$ (applicant pool) arriving in a uniformly random order, the goal is to select a set of at most k applicants in order to maximize a profit function $f : 2^S \mapsto \mathbb{R}$. We assume f is non-negative throughout this paper. For example, when $f(T)$ is the maximum individual value [17, 18], or when $f(T)$ is the sum of the individual values in T [30], the problem has been considered thoroughly in the literature. Indeed, both of these cases are special monotone non-negative submodular functions that we consider in this paper. A function $f : 2^S \mapsto \mathbb{R}$ is called *submodular* if and only if $\forall A, B \subseteq S : f(A) + f(B) \geq f(A \cup B) + f(A \cap B)$. An equivalent characterization is that the marginal profit of each item should be non-increasing, i.e., $f(A \cup \{a\}) - f(A) \leq f(B \cup \{a\}) - f(B)$ if $B \subseteq A \subseteq S$ and $a \in S \setminus B$. A function $f : 2^S \mapsto \mathbb{R}$ is *monotone* if and only if $f(A) \leq f(B)$ for $A \subseteq B \subseteq S$; it is *non-monotone* if is not necessarily the case. Since the number of sets is exponential, we assume a value oracle access to the submodular function; i.e., for a given set T, an algorithm can query an oracle to find its value $f(T)$. As we discuss below, maximizing a (monotone or non-monotone) submodular function which demonstrates economy of scale is a central and very general problem in combinatorial optimization and has been subject of a thorough study in the literature.

The closest in terms of generalization to our submodular multiple-choice secretary problem is the *matroid secretary problem* considered by Babaioff et al. [6]. In this problem, we are given a matroid by a ground set $\mathcal{U}$ of elements and a collection of independent (feasible) subsets $\mathcal{I} \subseteq 2^{\mathcal{U}}$ describing the sets of elements which can be simultaneously accepted. We recall that a matroid has three properties: 1) the empty set is independent; 2) every subset of an independent set is independent (closed under containment)[1]; and finally 3) if A and B are two independent sets and A has more elements than B, then there exists an element in A which is not in B and when added to B still gives an independent set[2]. The goal is to design online algorithms in which the structure of $\mathcal{U}$ and $\mathcal{I}$ is known at the outset (assume we have an oracle to answer whether a subset of $\mathcal{U}$ belongs to $\mathcal{I}$ or not), while the elements and their values are revealed one at a time in random order. As each element is presented, the algorithm must make an irrevocable decision to select or reject it such that the set of selected elements belongs to $\mathcal{I}$ at all times. Babaioff et al. present an $O(\log r)$-competitive algorithm for general matroids, where r is the rank of the matroid (the size of the maximal independent set), and constant-competitive algorithms for several special cases arising in practical scenarios including graphic matroids, truncated partition matroids, and bounded degree transversal matroids. However, they leave as a main open question the existence of constant-competitive algorithms for general matroids. Our constant-competitive algorithms for the submodular secretary problem in this paper can be considered in parallel with this open question. To generalize both results of Babaioff et al. and ours, we also consider the *submodular matroid secretary problem* in which we want to maximize a submodular function over all independent (feasible)

[1] This is sometimes called the *hereditary property*.

[2] This is sometimes called the *augmentation property* or the *independent set exchange property*.

subsets $\mathcal{I}$ of the given matroid. Moreover, we extend our approach to the case in which l matroids are given and the goal is to find the set of maximum value which is independent with respect to all the given matroids. We present an $O(l \log^2 r)$-competitive algorithm for the submodular matroid secretary problem generalizing previous results.

Prior to our work, there was no polynomial-time algorithm with a nontrivial guarantee for the case of l matroids—even in the offline setting—when l is not a fixed constant. Lee et al. [31] give a local-search procedure for the offline setting that runs in time $O(n^l)$ and achieves approximation ratio $l + \varepsilon$. Even the simpler case of having a linear function cannot be approximated to within a factor better than $\Omega(l/\log l)$ [25]. Our results imply an algorithm with guarantees $O(l \log r)$ and $O(l \log^2 r)$ for the offline and secretary settings, respectively. Both these algorithms run in time polynomial in l.

Our competitive ratio for the submodular secretary problem is $\frac{7}{1-1/e}$. Though our algorithm is relatively simple, it has several phases and its analysis is relatively involved. As we point out below, we cannot obtain any approximation factor better than $1 - 1/e$ even for offline special cases of our setting unless $\mathbf{P} = \mathbf{NP}$. A natural generalization of a submodular function while still preserving economy of scale is a subadditive function $f : 2^S \mapsto \mathbb{R}$ in which $\forall A, B \subseteq S : f(A) + f(B) \geq f(A \cup B)$. In this paper, we show that if we consider the subadditive secretary problem instead of the submodular secretary problem, there is no algorithm with competitive ratio $\tilde{o}(\sqrt{n})$. We complement this result by giving an $O(\sqrt{n})$-competitive algorithm for the subadditive secretary problem.

Background on submodular maximization Submodularity, a discrete analog of convexity, has played a central role in combinatorial optimization [32]. It appears in many important settings including cuts in graphs [27, 20, 34], plant location problems [9, 8], rank function of matroids [11], and set covering problems [12].

The problem of maximizing a submodular function is of essential importance, with special cases including Max Cut [20], Max Directed Cut [23], hypergraph cut problems, maximum facility location [1, 9, 8], and certain restricted satisfiability problems [24, 14]. While the Min Cut problem in graphs is a classical polynomial-time solvable problem, and more generally it has been shown that any submodular function can be minimized in polynomial time [27, 35], maximization turns out to be more difficult and indeed all the aforementioned special cases are NP-hard.

Max-k-Cover, where the goal is to choose k sets whose union is as large as possible, is another related problem. It is shown that a greedy algorithm provides a $(1 - 1/e)$-approximation for Max-k-Cover [29] and this is optimal unless $\mathbf{P} = \mathbf{NP}$ [12]. More generally, we can view this problem as maximization of a monotone submodular function under a cardinality constraint, that is, we seek a set S of size k maximizing $f(S)$. The greedy algorithm again provides a $(1 - 1/e)$-approximation for this problem [33]. A $1/2$-approximation has been developed for maximizing monotone submodular functions under a matroid constraint [16]. A $(1 - 1/e)$-approximation has been also obtained for a knapsack constraint [36], and for a special class of submodular functions under a matroid constraint [7].

Recently constant factor $(\frac{3}{4}+\varepsilon)$-approximation algorithms for maximizing non-negative non-monotone submodular functions has also been obtained [15]. Typical examples of such a problem are max cut and max directed cut. Here, the best approximation factors are 0.878 for max cut [20] and 0.859 for max directed cut [14]. The approximation factor for max cut has been proved optimal, assuming the Unique Games Conjecture [28]. Generalizing these results, Vondrak very recently obtains a constant factor approximation algorithm for maximizing non-monotone submodular functions under a matroid constraint [38]. Subadditive maximization has been also considered recently (e.g. in the context of maximizing welfare [13]).

Submodular maximization also plays a role in maximizing the difference of a monotone submodular function and a modular function. A typical example of this type is the maximum facility location problem in which we want to open a subset of facilities and maximize the total profit from clients minus the opening cost of facilities. Approximation algorithms have been developed for a variant of this problem which is a special case of maximizing nonnegative submodular functions [1, 9, 8]. The current best approximation factor known for this problem is 0.828 [1]. Asadpour et al. [3] study the problem of maximizing a submodular function in a stochastic setting, and obtain constant-factor approximation algorithms.

Our results and techniques The main theorem in this paper is as follows.

Theorem 1. *There exists a $\frac{7}{1-1/e}$-competitive algorithm for the monotone submodular secretary problem. More generally there exists a $8e^2$-competitive algorithm for the non-monotone submodular secretary problem.*

We prove Theorem 1 in Section 2. We first present our simple algorithms for the problem. Since our algorithm for the general non-monotone case uses that of monotone case, we first present the analysis for the latter case and then extend it for the former case. We divide the input stream into equal-sized segments, and show that restricting the algorithm to pick only one item from each segment decreases the value of the optimum by at most a constant factor. Then in each segment, we use a standard secretary algorithm to pick the best item conditioned on our previous choices. We next prove that these local optimization steps lead to a global near-optimal solution.

The argument breaks for the non-monotone case since the algorithm actually approximates a set which is larger than the optimal solution. The trick is to invoke a new structural property of (non-monotone) submodular functions which allows us to divide the input into two equal portions, and randomly solve the problem on one.

Indeed Theorem 1 can be extended for the submodular matroid secretary problem as follows.

Theorem 2. *There exists an $O(l \log^2 r)$ competitive algorithm for the (non-monotone) matroid submodular secretary problem, where r is the maximum rank of the given l matroids.*

We prove theorem 2 in Section 3. We note that in the submodular matroid secretary problem, selecting (bad) elements early in the process might prevent us from selecting (good) elements later since there are matroid independence (feasibility) constraints. To overcome this issue, we only work with the first half of the input. This guarantees that at each point in expectation there is a large portion of the optimal solution that can be added to our current solution without violating the matroid constraint. However, this set may not have a high value. As a remedy we prove there is a near-optimal solution all of whose large subsets have a high value. This novel argument may be of its own interest.

We shortly mention in Section 4 our results for maximizing a submodular secretary problem with respect to l knapsack constraints. In this setting, there are l knapsack capacities $C_i : 1 \leq i \leq l$, and each item j has different weights w_{ij} associated with each knapsack. A set T of items is feasible if and only if for *each* knapsack i, we have $\sum_{j \in T} w_{ij} \leq C_i$.

Theorem 3. *There exists an $O(l)$-competitive algorithm for the (non-monotone) multiple knapsack submodular secretary problem, where l denotes the number of given knapsack constraints.*

The only previous relevant work that we are aware of is that of Lee et al. [31] which gives a $(5+\varepsilon)$-approximation in the offline setting if l is a fixed constant. Our result gives a poorer guarantee, however, it works for any value of l.

We next show that indeed submodular secretary problems are the most general cases that we can hope for constant competitiveness.

Theorem 4. *For the subadditive secretary problem, there is no algorithm with competitive ratio in $\tilde{o}(\sqrt{n})$. However there is an algorithm with almost tight $O(\sqrt{n})$ competitive ratio in this case.*

We prove Theorem 4 in Section 5. The algorithm for the matching upper bound is very simple, however the lower bound uses clever ideas and indeed works in a more general setting. We construct a subadditive function, which interestingly is almost submodular, and has a "hidden good set". Roughly speaking, the value of any query to the oracle is proportional to the intersection of the query and the hidden good set. However, the oracle's response does not change unless the query has considerable intersection with the good set which is hidden. Hence, the oracle does not give much information about the hidden good set.

Finally in our concluding remarks in Section 6, we briefly discuss two other aggregate functions max and min, where the latter is not even submodular and models a bottle-neck situation in the secretary problem.

All omitted proofs can be found in the appendix.

2 The submodular secretary problem

2.1 Algorithms

In this sections, we present the algorithms used to prove Theorem 1. In the classic secretary problem, the efficiency value of each secretary is known only after she arrives. In order to marry this with the value oracle model, we say that the oracle answers the query regarding the efficiency of a set $S' \subseteq S$ only if all the secretaries in S' have already arrived and been interviewed.

Our algorithm for the monotone submodular case is relatively simple though its analysis is relatively involved. First we assume that n is a multiple of k, since otherwise we could virtually insert $n - k\lfloor \frac{n}{k} \rfloor$ dummy secretaries in the input: for any subset A of dummy secretaries and a set $B \subseteq S$, we have that $f(A \cup B) = f(B)$. In other words, there is no profit in employing the dummy secretaries. To be more precise, we simulate the augmented input

Algorithm 1 Monotone Submodular Secretary Algorithm

Input: A monotone submodular function $f : 2^S \mapsto \mathbb{R}$, and a randomly permuted stream of secretaries, denoted by $(a_1, a_2, \ldots, a_n)$, where n is an integer multiple of k.
Output: A subset of at most k secretaries.

Let $T_0 \leftarrow \emptyset$
Let $l \leftarrow n/k$
for $i \leftarrow 1$ **to** k **do** {phase i}
  Let $u_i \leftarrow (i-1)l + l/e$
  Let $\alpha_i \leftarrow \max_{(i-1)l \leq j < u_i} f(T_{i-1} \cup \{a_j\})$
  if $\alpha_i < f(T_{i-1})$ **then**
    $\alpha_i \leftarrow f(T_{i-1})$
  end if
  Pick an index $p_i : u_i \leq p_i < il$ such that $f(T_{i-1} \cup \{a_{p_i}\}) \geq \alpha_i$
  if such an index p_i exists **then**
    Let $T_i \leftarrow T_{i-1} \cup \{a_{p_i}\}$
  else
    Let $T_i \leftarrow T_{i-1}$
  end if
end for
Output T_k as the solution

in such a way that these secretaries are arriving uniformly at random similarly to the real ones. Thus, we say that n is a multiple of k without loss of generality.

We partition the input stream into k equally-sized segments, and, roughly speaking, try to employ the *best* secretary in each segment. Let $l := \frac{n}{k}$ denote the length of each segment. Let $a_1, a_2, \cdots, a_n$ be the actual ordering in which the secretaries are interviewed. Then, we have the following segments.

$$\begin{aligned} S_1 &= \{a_1, a_2, \ldots, a_l\}, \\ S_2 &= \{a_{l+1}, a_{l+2}, \ldots, a_{2l}\}, \\ &\vdots \\ S_k &= \{a_{(k-1)l+1}, a_{(k-1)l+2}, \ldots, a_n\}. \end{aligned}$$

We employ at most one secretary from each segment S_i. Note that this way of having several phases for the secretary problem seems novel in this paper, since in previous works there are usually only two phases (see e.g. [21]). The phase i of our algorithm corresponds to the time interval when the secretaries in S_i arrive. Let T_i be the set of secretaries that we have employed from $\bigcup_{j=1}^{i} S_j$. Define $T_0 := \emptyset$ for convenience. In phase i, we try to employ a secretary e from S_i that maximizes $f(T_{i-1} \cup \{e\}) - f(T_{i-1})$. For each $e \in S_i$, we define $g_i(e) = f(T_{i-1} \cup \{e\}) - f(T_{i-1})$. Then, we are trying to employ a secretary $x \in S_i$ that has the maximum value for $g_i(e)$. Using a classic algorithm for the *secretary problem* (see [10] for instance) for employing the single secretary, we can solve this problem with constant probability $1/e$. Hence, with constant probability, we pick the secretary that maximizes our local profit in each phase. It leaves us to prove that this local optimization leads to a reasonable global guarantee.

The previous algorithm fails in the non-monotone case. Observe that the first **if** statement is never true for a monotone function, however, for a non-monotone function this guarantees the values of sets T_i are non-decreasing. Algorithm 2 first divides the input stream into two equal-sized parts: U_1 and U_2. Then, with probability $1/2$, it calls Algorithm 1 on U_1, whereas with the same probability, it skips over the first half of the input, and runs Algorithm 1 on U_2.

2.2 Analysis

In this section, we prove Theorem 1. Since the algorithm for the non-monotone submodular secretary problem uses that for the monotone submodular secretary problem, first we start with the monotone case.

Algorithm 2 Submodular Secretary Algorithm

Input: A (possibly non-monotone) submodular function $f : 2^S \mapsto \mathbb{R}$, and a randomly permuted stream of secretaries, denoted by $(a_1, a_2, \ldots, a_n)$, where n is an integer multiple of $2k$.
Output: A subset of at most k secretaries.

```
Let U_1 := {a_1, a_2, ..., a_⌊n/2⌋}
Let U_2 := {a_⌊n/2⌋ + 1, ..., a_{n-1}, a_n}
Let 0 ≤ X ≤ 1 be a uniformly random value.
if X ≤ 1/2 then
   Run Algorithm 1 on U_1 to get S_1
   Output S_1 as the solution
else
   Run Algorithm 1 on U_2 to get S_2
   Output S_2 as the solution
end if
```

2.2.1 Monotone submodular

We prove in this section that for Algorithm 1, the expected value of $f(T_k)$ is within a constant factor of the optimal solution. Let $R = \{a_{i_1}, a_{i_2}, \cdots, a_{i_k}\}$ be the optimal solution. Note that the set $\{i_1, i_2, \cdots, i_k\}$ is a uniformly random subset of $\{1, 2, \cdots, n\}$ with size k. It is also important to note that the permutation of the elements of the optimal solution on these k places is also uniformly random, and is independent from the set $\{i_1, i_2, \cdots, i_k\}$. For example, any of the k elements of the optimum can appear as a_{i_1}. These are two key facts used in the analysis.

Before starting the analysis, we present a simple property of submodular functions which will prove useful in the analysis. The proof of the lemma is standard, and is included in the appendix for the sake of completeness.

Lemma 5. *If $f : 2^S \mapsto \mathbb{R}$ is a submodular function, we have $f(B) - f(A) \leq \sum_{a \in B \setminus A} [f(A \cup \{a\}) - f(A)]$ for any $A \subseteq B \subseteq S$.*

Define $\mathcal{X} := \{S_i : |S_i \cap R| \neq \emptyset\}$. For each $S_i \in \mathcal{X}$, we pick one element, say s_i, of $S_i \cap R$ randomly. These selected items form a set called $R' = \{s_1, s_2, \cdots, s_{|\mathcal{X}|}\} \subseteq R$ of size $|\mathcal{X}|$. Since our algorithm approximates such a set, we study the value of such random samples of R in the following lemmas. We first show that restricting ourselves to picking at most one element from each segment does not prevent us from picking many elements from the optimal solution (i.e., R).

Lemma 6. *The expected value of the number of items in R' is at least $k(1 - 1/e)$.*

Proof. We know that $|R'| = |\mathcal{X}|$, and $|\mathcal{X}|$ is equal to k minus the number of sets S_i whose intersection with R is empty. So, we compute the expected number of these sets, and subtract this quantity from k to obtain the expected value of $|\mathcal{X}|$ and thus $|R'|$.

Consider a set S_q, $1 \leq q \leq k$, and the elements of $R = \{a_{i_1}, a_{i_2}, \ldots, a_{i_k}\}$. Define $\mathcal{E}_j$ as the event that a_{i_j} is not in S_q. We have $\Pr(\mathcal{E}_1) = \frac{(k-1)l}{n} = 1 - \frac{1}{k}$, and for any $i : 1 < i \leq k$, we get

$$\Pr\left(\mathcal{E}_i \middle| \bigcap_{j=1}^{i-1} \mathcal{E}_j\right) = \frac{(k-1)l - (i-1)}{n - (i-1)} \leq \frac{(k-1)l}{n} = 1 - \frac{1}{k},$$

where the last inequality follows from a simple mathematical fact: $\frac{x-c}{y-c} \leq \frac{x}{y}$ if $c \geq 0$ and $x \leq y$. Now we conclude that the probability of the event $S_q \cap R = \emptyset$ is

$$\Pr(\cap_{i=1}^{k} \mathcal{E}_i) = \Pr(\mathcal{E}_1) \cdot \Pr(\mathcal{E}_2 | \mathcal{E}_1) \cdots \Pr(\mathcal{E}_k | \cap_{j=1}^{k-1} \mathcal{E}_j) \leq \left(1 - \frac{1}{k}\right)^k \leq \frac{1}{e}.$$

Thus each of the sets $S_1, S_2, \ldots, S_k$ does not intersect with R with probability at most $1/e$. Hence, the expected number of such sets is at most k/e. Therefore, the expected value of $|\mathcal{X}| = |R'|$ is at least $k(1 - 1/e)$. ∎

The next lemma materializes the proof of an intuitive statement: if you randomly sample elements of the set R, you expect to obtain a profit proportional to the size of your sample. An analog this is proved in [13] for the case when $|R|/|A|$ is an integer.

Lemma 7. *For a random subset A of R, the expected value of $f(A)$ is at least $\frac{|A|}{k} \cdot f(R)$.*

Proof. Let $(x_1, x_2, \ldots, x_k)$ be a random ordering of the elements of R. For $r = 1, 2, \ldots, k$, let F_r be the expectation of $f(\{x_1, \ldots, x_r\})$, and define $D_k := F_k - F_{k-1}$, where F_0 is interpreted to be equal to zero. Letting $a := |A|$, note that $f(R) = F_k = D_1 + \cdots + D_k$, and that the expectation of $f(A)$ is equal to $F_a = D_1 + \cdots + D_a$. We claim that $D_1 \geq D_2 \geq \cdots \geq D_k$, from which the lemma follows easily. Let $(y_1, y_2, \ldots, y_k)$ be a cyclic permutation of $(x_1, x_2, \ldots, x_k)$, where $y_1 = x_k, y_2 = x_1, y_3 = x_2, \ldots, y_k = x_{k-1}$. Notice that for $i < k$, F_i is equal to the expectation of $f(\{y_2, \ldots, y_{i+1}\})$ since $\{y_2, \ldots, y_{i+1}\}$ is equal to $\{x_1, \ldots, x_i\}$.

F_i is also equal to the expectation of $f(\{y_1, \ldots, y_i\})$, since the sequence $(y_1, \ldots, y_i)$ has the same distribution as that of $(x_1, \cdots, x_i)$. Thus, D_{i+1} is the expectation of $f(\{y_1, \ldots, y_{i+1}\}) - f(\{y_2, \ldots, y_{i+1}\})$, whereas D_i is the expectation of $f(\{y_1, \ldots, y_i\}) - f(\{y_2, \ldots, y_i\})$. The submodularity of f implies that $f(\{y_1, \ldots, y_{i+1}\}) - f(\{y_2, \ldots, y_{i+1}\})$ is less than or equal to $f(\{y_1, \ldots, y_i\}) - f(\{y_2, \ldots, y_i\})$, hence $D_{i+1} \leq D_i$. $\square$

Here comes the crux of our analysis where we prove that the local optimization steps (i.e., trying to make the best move in each segment) indeed lead to a globally approximate solution.

Lemma 8. *The expected value of $f(T_k)$ is at least $\frac{|R'|}{7k} \cdot f(R)$.*

Proof. Define $m := |R'|$ for the ease of reference. Recall that R' is a set of secretaries $\{s_1, s_2, \ldots, s_m\}$ such that $s_i \in S_{h_i} \cap R$ for $i : 1 \leq i \leq m$ and $h_i : 1 \leq h_i \leq k$. Also assume without loss of generality that $h_{i'} \leq h_i$ for $1 \leq i' < i \leq m$, for instance, s_1 is the first element of R' to appear. Define Δ_j for each $j : 1 \leq j \leq k$ as the gain of our algorithm while working on the segment S_j. It is formally defined as $\Delta_j := f(T_j) - f(T_{j-1})$. Note that due to the first **if** statement in the algorithm, $\Delta_j \geq 0$ and thus $\mathbf{E}[\Delta_j] \geq 0$. With probability $1/e$, we choose the element in S_j which maximizes the value of $f(T_j)$ (given that the set T_{j-1} is fixed). Notice that by definition of R' only one s_i appears in S_{h_i}. Since $s_i \in S_{h_i}$ is one of the options,

$$\mathbf{E}[\Delta_{h_i}] \geq \frac{\mathbf{E}[f(T_{h_i-1} \cup \{s_i\}) - f(T_{h_i-1})]}{e}. \tag{1}$$

To prove by contradiction, suppose $\mathbf{E}[f(T_k)] < \frac{m}{7k} \cdot f(R)$. Since f is monotone, $\mathbf{E}[f(T_j)] < \frac{m}{7k} \cdot f(R)$ for any $0 \leq j \leq k$. Define $B := \{s_i, s_{i+1}, \cdots, s_m\}$. By Lemma 5 and monotonicity of f,

$$f(B) \leq f(B \cup T_{h_i-1}) \leq f(T_{h_i-1}) + \sum_{j=i}^{m} [f(T_{h_i-1} \cup \{s_j\}) - f(T_{h_i-1})],$$

which implies

$$\mathbf{E}[f(B)] \leq \mathbf{E}[f(T_{h_i-1})] + \sum_{j=i}^{m} \mathbf{E}[f(T_{h_i-1} \cup \{s_j\}) - f(T_{h_i-1})].$$

Since the items in B are distributed uniformly at random, and there is no difference between s_{i_1} and s_{i_2} for $i \leq i_1, i_2 \leq m$, we can say

$$\mathbf{E}[f(B)] \leq \mathbf{E}[f(T_{h_i-1})] + (m - i + 1) \cdot \mathbf{E}[f(T_{h_i-1} \cup \{s_i\}) - f(T_{h_i-1})]. \tag{2}$$

We conclude from (1) and (2)

$$\mathbf{E}[\Delta_{h_i}] \geq \frac{\mathbf{E}[f(T_{h_i-1} \cup \{s_i\}) - f(T_{h_i-1})]}{e} \geq \frac{\mathbf{E}[f(B)] - \mathbf{E}[f(T_{h_i-1})]}{e(m - i + 1)}.$$

Since B is a random sample of R, we can apply Lemma 7 to get $\mathbf{E}[f(B)] \geq \frac{|B|}{k} f(R) = f(R)(m-i+1)/k$. Since $\mathbf{E}[f(T_{h_i-1})] \leq \frac{m}{7k} \cdot f(R)$, we reach

$$\mathbf{E}[\Delta_{h_i}] \geq \frac{\mathbf{E}[f(B)] - \mathbf{E}[f(T_{h_i-1})]}{e(m-i+1)} \geq \frac{f(R)}{ek} - \frac{m}{7k} f(R) \cdot \frac{1}{e(m-i+1)}. \tag{3}$$

Adding up (3) for $i : 1 \leq i \leq \lceil m/2 \rceil$, we obtain

$$\sum_{i=1}^{\lceil m/2 \rceil} \mathbf{E}[\Delta_{h_i}] \geq \left\lceil \frac{m}{2} \right\rceil \cdot \frac{f(R)}{ek} - \frac{m}{7ek} \cdot f(R) \cdot \sum_{i=1}^{\lceil m/2 \rceil} \frac{1}{m-i+1}.$$

Since $\sum_{j=a}^{b} \frac{1}{j} \leq \ln \frac{b}{a+1}$ for any integer values of $a, b : 1 < a \leq b$, we conclude

$$\sum_{i=1}^{\lceil m/2 \rceil} \mathbf{E}[\Delta_{h_i}] \geq \left\lceil \frac{m}{2} \right\rceil \cdot \frac{f(R)}{ek} - \frac{m}{7ek} \cdot f(R) \cdot \ln \frac{m}{\lfloor \frac{m}{2} \rfloor}.$$

A similar argument for the range $1 \leq i \leq \lfloor m/2 \rfloor$ gives

$$\sum_{i=1}^{\lfloor \frac{m}{2} \rfloor} \mathbf{E}[\Delta_{h_i}] \geq \left\lfloor \frac{m}{2} \right\rfloor \cdot \frac{f(R)}{ek} - \frac{m}{7ek} \cdot f(R) \cdot \ln \frac{m}{\lceil \frac{m}{2} \rceil}.$$

We also know that both $\sum_{i=1}^{\lfloor m/2 \rfloor} \mathbf{E}[\Delta_{h_i}]$ and $\sum_{i=1}^{\lceil m/2 \rceil} \mathbf{E}[\Delta_{h_i}]$ are at most $\mathbf{E}[f(T_k)]$ because $f(T_k) \geq \sum_{i=1}^{m} \Delta_{h_i}$. We conclude with

$$\begin{aligned}
2\mathbf{E}[f(T_k)] &\geq \left\lceil \frac{m}{2} \right\rceil \frac{f(R)}{ek} - \frac{mf(R)}{7ek} \cdot \ln \frac{m}{\lfloor \frac{m}{2} \rfloor} + \left\lfloor \frac{m}{2} \right\rfloor \frac{f(R)}{ek} - \frac{mf(R)}{7ek} \cdot \ln \frac{m}{\lceil \frac{m}{2} \rceil} \\
&\geq \frac{mf(R)}{ek} - \frac{mf(R)}{7ek} \cdot \ln \frac{m^2}{\lfloor \frac{m}{2} \rfloor \lceil \frac{m}{2} \rceil}, \qquad \text{and since } \frac{m^2}{\lfloor m/2 \rfloor \lceil m/2 \rceil} < 4.5 \\
&\geq \frac{mf(R)}{ek} - \frac{mf(R)}{7ek} \cdot \ln 4.5 = \frac{mf(R)}{k} \cdot \left(\frac{1}{e} - \frac{\ln 4.5}{7e} \right) \geq \frac{mf(R)}{k} \cdot \frac{2}{7},
\end{aligned}$$

which contradicts $\mathbf{E}[f(T_k)] < \frac{mf(R)}{7k}$, hence proving the supposition false. □

The following theorem wraps up the analysis of the algorithm.

Theorem 9. *The expected value of the output of our algorithm is at least* $\frac{1-1/e}{7} f(R)$.

Proof. The expected value of $|R'| = m \geq (1-1/e)k$ from Lemma 6. In other words, we have $\sum_{m=1}^{k} \Pr[|R'| = m] \cdot m \geq \left(1 - \frac{1}{e}\right) k$. We know from Lemma 8 that if the size of R' is m, the expected value of $f(T_k)$ is at least $\frac{m}{7k} f(R)$, implying that $\sum_{v \in V} \Pr\left[f(T_k) = v \middle| |R'| = m\right] \cdot v \geq \frac{m}{7k} f(R)$, where V denotes the set of different values that $f(T_k)$ can get. We also know that

$$\begin{aligned}
\mathbf{E}[f(T_k)] &= \sum_{m=1}^{k} \mathbf{E}[f(T_k) | |R'| = m] \Pr[|R'| = m] \geq \sum_{m=1}^{k} \frac{m}{7k} f(R) \Pr[|R'| = m] \\
&= \frac{f(R)}{7k} \mathbf{E}[|R'|] \geq \frac{1-1/e}{7} f(R).
\end{aligned}$$ □

2.2.2 Non-monotone submodular

Before starting the analysis of Algorithm 2 for non-monotone functions, we show an interesting property of Algorithm 1. Consistently with the notation of Section 2.2, we use R to refer to some optimal solution. Recall that we partition the input stream into (almost) equal-sized segments $S_i : 1 \leq i \leq k$, and pick one item from each. Then T_i denotes the set of items we have picked at the completion of segment i. We show that $f(T_k) \geq \frac{1}{2e} f(R \cup T_i)$ for some integer i, even when f is not monotone. Roughly speaking, the proof mainly follows from the submodularity property and Lemma 5.

Lemma 10. *If we run the monotone algorithm on a (possibly non-monotone) submodular function f, we obtain $f(T_k) \geq \frac{1}{2e^2} f(R \cup T_i)$ for some i.*

Proof. Consider the stage $i+1$ in which we want to pick an item from S_{i+1}. Lemma 5 implies

$$f(R \cup T_i) \leq f(T_i) + \sum_{a \in R \setminus T_i} f(T_i \cup \{a\}) - f(T_i).$$

At least one of the two right-hand side terms has to be larger than $f(R \cup T_i)/2$. If this happens to be the first term for any i, we are done: $f(T_k) \geq f(T_i) \geq \frac{1}{2} f(R \cup T_i)$ since $f(T_k) \geq f(T_i)$ by the definition of the algorithm: the first **if** statement makes sure $f(T_i)$ values are non-decreasing. Otherwise assume that the lower bound occurs for the second terms for all values of i.

Consider the events that among the elements in $R \setminus T_i$ exactly one, say a, falls in S_{i+1}. Call this event $\mathcal{E}_a$. Conditioned on $\mathcal{E}_a$, $\Delta_{i+1} := f(T_{i+1}) - f(T_i)$ is at least $f(T_i \cup \{a\}) - f(T_i)$ with probability $1/e$: i.e., if the algorithm picks the best secretary in this interval. Each event $\mathcal{E}_a$ occurs with probability at least $\frac{1}{k} \cdot \frac{1}{e}$. Since these events are disjoint, we have

$$\mathbf{E}[\Delta_{i+1}] \geq \sum_{a \in R \setminus T_i} \Pr[\mathcal{E}_a] \cdot \frac{1}{e}[f(T_{i+1}) - f(T_i)] \geq \frac{1}{e^2 k} \sum_{a \in R \setminus T_i} f(T_i \cup \{a\}) - f(T_i) \geq \frac{1}{2e^2 k} f(R \cup T_i),$$

and by summing over all values of i, we obtain

$$\mathbf{E}[f(T_k)] = \sum_i \mathbf{E}[\Delta_i] \geq \sum_i \frac{1}{2e^2 k} f(R \cup T_i) \geq \frac{1}{2e^2} \min_i f(R \cup T_i). \qquad \square$$

Unlike the case of monotone functions, we cannot say that $f(R \cup T_i) \geq f(R)$, and conclude that our algorithm is constant-competitive. Instead, we need to use other techniques to cover the cases that $f(R \cup T_i) < f(R)$. The following lemma presents an upper bound on the value of the optimum.

Lemma 11. *For any pair of disjoint sets Z and Z', and a submodular function f, we have $f(R) \leq f(R \cup Z) + f(R \cup Z')$.*

We are now at a position to prove the performance guarantee of our main algorithm.

Theorem 12. *Algorithm 2 has competitive ratio $8e^2$.*

Proof. Let the outputs of the two algorithms be sets Z and Z', respectively. The expected value of the solution is thus $[f(Z) + f(Z')]/2$.

We know that $\mathbf{E}[f(Z)] \geq c' f(R \cup X_1)$ for some constant c', and $X_1 \subseteq U_1$. The only difference in the proof is that each element of $R \setminus Z$ appears in the set S_i with probability $1/2k$ instead of $1/k$. But we can still prove the above lemma for $c' := 1/4e^2$. Same holds for Z': $\mathbf{E}[f(Z')] \geq \frac{1}{4e} f(R \cup X_2)$ for some $X_2 \subseteq U_2$.

Since U_1 and U_2 are disjoint, so are X_1 and X_2. Hence, the expected value of our solution is at least $\frac{1}{4e^2}[f(R \cup X_1) + f(R \cup X_2)]/2$, which via Lemma 11 is at least $\frac{1}{8e^2} f(R)$. $\square$

3 The submodular matroid secretary problem

In this section, we prove Theorem 2. We first design an $O(\log^2 r)$-competitive algorithm for maximizing a monotone submodular function, when there are matroid constraints for the set of selected items. Here we are allowed to choose a subset of items only if it is an independent set in the given matroid.

The matroid $(\mathcal{U}, \mathcal{I})$ is given by an oracle access to $\mathcal{I}$. Let n denote the number of items, i.e., $n := |\mathcal{U}|$, and r denotes the rank of the matroid. Let $S \in \mathcal{I}$ denote an optimal solution that maximizes the function f. We focus our analysis on a refined set $S^* \subseteq S$ that has certain nice properties: 1) $f(S^*) \geq (1 - 1/e)f(S)$, and 2) $f(T) \geq f(S^*)/\log r$ for any $T \subseteq S^*$ such that $|T| = \lfloor |S^*|/2 \rfloor$. We cannot necessarily find S^*, but we prove that such a set exists.

Start by letting $S^* = S$. As long as there is a set T violating the second property above, remove T from S^*, and continue. The second property clearly holds at the termination of the procedure. In order to prove the first property, consider one iteration. By submodularity (subadditivity to be more precise) we have $f(S^* \setminus T) \geq f(S^*) - f(T) \geq (1 - 1/\log r)f(S^*)$. Since each iteration halves the set S^*, there are at most $\log r$ iterations. Therefore, $f(S^*) \geq (1 - 1/\log r)^{\log r} \cdot f(S) \geq (1 - 1/e)f(S)$.

We analyze the algorithm assuming the parameter $|S^*|$ is given, and achieve a competitive ratio $O(\log r)$. If $|S^*|$ is unknown, though, we can guess its value (from a pool of $\log r$ different choices) and continue with Lemma 13. This gives an $O(\log^2 r)$-competitive ratio.

Lemma 13. *Given $|S^*|$, Algorithm 3 (presented in Appendix B) picks an independent subset of items with size $|S^*|/2$ whose expected value is at least $f(S^*)/4e\log r$.*

Proof. Let $k := |S^*|$. We divide the input stream of n items into k segments of (almost) equal size. We only pick $k/2$ items, one from each of the first $k/2$ segments.

Similarly to Algorithm 1 for the submodular secretary problem, when we work on each segment, we try to pick an item that maximizes the marginal value of the function given the previous selection is fixed (see the **for** loop in Algorithm 1). We show that the expected gain in each of the first $k/2$ segments is at least a constant fraction of $f(S^*)/k\log r$.

Suppose we are working on segment $i \leq k/2$, and let Z be the set of items already picked; so $|Z| \leq i - 1$. Furthermore, assume $f(Z) \leq f(S^*)/2\log r$ since otherwise, the lemma is already proved. By matroid properties we know there is a set $T \subseteq S^* \setminus Z$ of size $\lfloor k/2 \lfloor$ such that $T \cup Z \in \mathcal{I}$. The second property of S^* gives $f(T) \geq f(S^*)/\log r$.

From Lemma 5 and monotonicity of f, we obtain

$$\sum_{s \in T} [f(Z \cup \{s\}) - f(Z)] \geq f(T \cup Z) - f(Z) \geq f(T) - f(Z) \geq f(S^*)/2\log r.$$

Note that each item in T appears in this segment with probability $2/k$ because we divided the input stream into $k/2$ equal segments. Since in each segment we pick the item giving the maximum marginal value with probability $1/e$, the expected gain in this segment is at least

$$\sum_{s \in T} \frac{1}{e} \cdot \frac{2}{k} \cdot [f(Z \cup \{s\}) - f(Z)] \geq f(S^*)/ek\log r.$$

We have this for each of the first $k/2$ segments, so the expected value of our solution is at least $f(S^*)/2e\log r$. $\square$

Finally, it is straightforward (and hence the details are omitted) to combine the algorithm in this section with Algorithm 2 for the nonmonotone submodular secretary problem, to obtain an $O(\log^2 r)$-competitive algorithm for the non-monotone submodular secretary problem subject to a matroid constraint.

Here we show the same algorithm works when there are $l \geq 1$ matroid constraints and achieves a competitive ratio of $O(l \log^2 r)$. We just need to respect all matroid constraints in Algorithm 3. This finishes the proof of Theorem 2.

Lemma 14. *Given $|S^*|$, Algorithm 3 (presented in Appendix B) picks an independent subset of items (i.e., independent with respect to all matroids) with expected value at least $f(S^*)/4el\log r$.*

Proof. The proof is similar to the proof of Lemma 13. We show that the expected gain in each of the first $k/2l$ segments is at least a constant fraction of $f(S^*)/k \log r$.

Suppose we are working on segment $i \le k/2l$, and let Z be the set of items already picked; so $|Z| \le i-1$. Furthermore, assume $f(Z) \le f(S^*)/2 \log r$ since otherwise, the lemma is already proved. We claim that there is a set $T \subseteq S^* \setminus Z$ of size $k - l \times \lfloor k/2l \rfloor \ge k/2$ such that $T \cup Z$ is an independent set in all matroids. The proof is as follows. We know that there exists a set $T_1 \subseteq S^*$ whose union with Z is an independent set of the first matroid, and the size of T_1 is at least $|S^*| - |Z|$. This can be proved by the exchange property of matroids, i.e., adding Z to the independent set S^* does not remove more than $|Z|$ items from S^*. Since T_1 is independent with respect to the second matroid (as it is a subset of S^*), we can prove that there exists a set $T_2 \subseteq T_1$ of size at least $|T_1| - |Z|$ such that $Z \cup T_2$ is an independent set in the second matroid. If we continue this process for all matroid constraints, we can prove that there is a set T_l which is an independent set in all matroids, and has size at least $|S^*| - l|Z| \ge k - l \times \lfloor k/2l \rfloor \ge k/2$ such that $Z \cup T_l$ is independent with respect to all the given matroids. The rest of the proof is similar to the proof of Lemma 13—we just need to use the set T_l instead of the set T in the proof.

Since we are gaining a constant times $f(S^*)/k \log r$ in each of the first $k/2l$ segments, the expected value of the final solution is at least a constant times $f(S^*)/l \log r$. □

4 Knapsack constraints

In this section, we prove Theorem 3. We first outline how to reduce an instance with multiple knapsacks to an instance with only one knapsack, and then we show how to solve the single knapsack instance.

Without loss of generality, we can assume that all knapsack capacities are equal to one. Let I be the given instance with the value function f, and item weights w_{ij} for $1 \le i \le l$ and $1 \le j \le n$. Define a new instance I' with one knapsack of capacity one in which the weight of the item j is $w'_j := \max_i w_{ij}$. We first prove that this reduction loses no more than a factor $4l$ in the total value. Take note that both the scaling and the weight transformation can be carried in an online manner as the items arrive. Hence, the results of this section hold for the online as well as the offline setting.

Lemma 15. *With instance I' defined above, we have $\frac{1}{4l}\operatorname{OPT}(I) \le \operatorname{OPT}(I') \le \operatorname{OPT}(I)$.*

Proof. The latter inequality is very simple: Take the optimal solultion to I'. This is also feasible in I since all the item weights in I are bounded by the weight in I'.

We next prove the other inequality. Let T be the optimal solution of I. An item j is called *fat* if $w'_j \ge 1/2$. Notice that there can be at most $2l$ fat items in T since $\sum_{j \in T} w'_j \le \sum_{j \in T} \sum_i w_{ij} \le l$. If there is any fat item with value at least $\operatorname{OPT}(I)/4l$, the statement of the lemma follows immediately, so we assume this is not the case. The total value of the fat items, say F, is at most $\operatorname{OPT}(I)/2$. Submodularity and non-negativity of f gives $f(T \setminus F) \ge f(T) - f(F) \ge \operatorname{OPT}(I)/2$. Sort the non-fat items in decreasing order of their value density (i.e., ratio of value to weight), and let T' be a maximal prefix of this ordering that is feasible with respect to I'. If $T' = T \setminus F$, we are done; otherwise, T' has weight at least $1/2$. Let x be the total weight of items in T' and let y indicate the total weight of item $T \setminus (F \cup T')$. Let α_x and α_y denote the densities of the two corresponding subsets of the items, respectively. Clearly $x + y \le l$ and $\alpha_x \ge \alpha_y$. Thus, $f(T \setminus F) = \alpha_x \cdot x + \alpha_y \cdot y \le \alpha_x(x+y) \le \alpha_x \cdot l$. Now $f(T') \ge \alpha_x \cdot \frac{1}{2} \ge \frac{1}{2l} f(T \setminus F) \ge \frac{1}{4l} f(T)$ finishes the proof. □

Here we show how to find a constant competitive algorithm when there is only one knapsack constraint. Let w_j denote the weight of item $j : 1 \le j \le n$, and assume without loss of generality that the capacity of the knapsack is 1. Moreover, let f be the value function which is a non-monotone submodular function. Let T be the optimal solution, and define $\operatorname{OPT} := f(T)$. The value of the parameter $\lambda \ge 1$ will be fixed below. Define T_1 and T_2 as the subset of T that appears in the first and second half of the input stream, respectively. We prove that if the value of each item is at most $\operatorname{OPT}/\lambda$, for sufficiently large λ, the random variable $|f(T_1) - f(T_2)|$ is bounded by $\operatorname{OPT}/2$ with a constant probability.

Each item of T goes to either T_1 or T_2 with probability $1/2$. Let the random variable X^1_j denote the increase of the value of $f(T_1)$ due to the possible addition of item j. Similarly X^2_j is defined for the same effect on $f(T_2)$. The two variables X^1_j and X^2_j have the same probability distribution, and because of submodularity and the fact that the value of item j is at most OPT/λ, the contribution of item j in $f(T_1) - f(T_2)$ can be seen as a random variable that always take values in range $[-\operatorname{OPT}/\lambda, \operatorname{OPT}/\lambda]$ with mean zero. (In fact, we also use the fact that in an optimal solution, the marginal value of any item is non-negative. Submodularity guarantees that this holds with respect to any of the subsets of T as well.) Azuma's inequality ensures that with constant probability the value of $|f(T_1) - f(T_2)|$ is not more than $\max\{f(T_1), f(T_2)\}/2$ for sufficiently large λ. Since both $f(T_1)$ and $f(T_2)$ are at most OPT, we can say that they are both at least $\mathrm{OPT}/4$, with constant probability.

The algorithm is as follows. Without loss of generality assume that all items are feasible, i.e., any one item fits into the knapsack. We flip a coin, and if it turns up "heads," we simply try to pick the one item with the maximum value. We do the following if the coin turns up "tails." We do not pick any items from the first half of the stream. Instead, we compute the maximum value set in the first half with respect to the knapsack constraint; Lee et al. give a constant fator approximation for this task. From the above argument, we know that $f(T_1)$ is at least $OPT/4$ since all the items have limited value in this case (i.e., at most OPT/λ). Therefore, we obtain a constant factor estimation of OPT by looking at the first half of the stream: i.e., if the estimate is $\hat{\mathrm{OPT}}$, we get $\mathrm{OPT}/c \le \hat{\mathrm{OPT}} \le \mathrm{OPT}$. After obtaining this estimate, we go over the second half of the input, and pick an item j if and only if it is feasible to pick this item, and moreover, the ratio of its marginal value to w_j is at least $\hat{\mathrm{OPT}}/6$.

Lemma 16. *The above algorithm is a constant competitive algorithm for the non-monotone submodular secretary problem with one knapsack constraint.*

Proof. We give the proof for the monotone case. Extending it for the non-monotone requires the same idea as was used in the proof of Theorem 2. First suppose there is an item with value at least OPT/λ. With probability $1/2$, we try to pick the best item, and we succeed with probability $1/e$. Thus, we get an $O(1)$ competitive ratio in this case.

In the other case, all the items have small contributions to the solution, i.e., less than OPT/λ. In this case, with constant probability, both $f(T_1)$ and $f(T_2)$ are at least $\mathrm{OPT}/4$. Hence, $\hat{\mathrm{OPT}}$ is a constant estimate for OPT. Let T' be the set of items picked by the algorithm in this case. If the sum of the weights of the items in T' is at least $1/2$, we are done, because all these items have (marginal) value density at least $\hat{\mathrm{OPT}}/6$, so $f(T') \ge (1/2)\cdot(\hat{\mathrm{OPT}}/6) = \hat{\mathrm{OPT}}/12 \ge \mathrm{OPT}/48$.

Otherwise, the total weight of T' is less than $1/2$. Therefore, there are items in T_2 that are not picked. There might be two reasons for this. There was not enough room in the knapsack, which means that the weight of the items in T_2 is more than $1/2$. However, there cannot be more than one such item in T_2, and the value of this item is not more than OPT/λ. Let z be this single big item, for future reference. Therefore, $f(T') \ge f(T_2) - \mathrm{OPT}/\lambda$ in this case.

The other case is when the ratios of some items from T_2 are less than $\hat{\mathrm{OPT}}/6$, and thus we do not pick them. Since they are all in T_2, their total weight is at most 1. Because of submodularity, the total loss due to these missed items is at most $\hat{\mathrm{OPT}}/6$. Submodularity and non-negativity of f then gives $f(T') \ge f(T_2) - f(\{z\}) - \hat{\mathrm{OPT}}/6 \ge \hat{\mathrm{OPT}} - \mathrm{OPT}\,\lambda - \hat{\mathrm{OPT}}/6 = O(\mathrm{OPT})$. □

5 The subadditive secretary problem

In this section, we prove Theorem 4 by presenting first a hardness result for approximation subadditive functions in general. The result applies in particular to our online setting. Surprisingly, the monotone subadditive function that we use here is *almost submodular*; see Proposition 19 below. Hence, our constant competitive ratio for submodular functions is nearly the most general we can achieve.

Definition 1 (Subadditive function maximization)**.** *Given a nonnegative subadditive function f on a ground set U, and a positive integer $k \le |U|$, the goal is to find a subset S of U of size at most k so as to maximize $f(S)$. The function f is accessible through a value oracle.*

5.1 Hardness result

In the following discussion, we assume that there is an upper bound of m on the size of sets given to the oracle. We believe this restriction can be lifted. If the function f is not required to be monotone, this is quite easy to have: simply let the value of the function f be zero for queries of size larger than m. Furthermore, depending on how we define the online setting, this may not be an *additional* restriction here. For example, we may not be able to query the oracle with secretaries that have already been rejected.

The main result of the section is the following theorem. It shows the subadditive function maximization is difficult to approximate, even in the offline setting.

Theorem 17. *There is no polynomial time algorithm to approximate an instance of subadditive function maximization within $\tilde{O}(\sqrt{n})$ of the optimum. Furthermore, no algorithm with exponential time 2^t can achieve an approximation ratio better than $\tilde{O}(\sqrt{n/t})$.*

First, we are going to define our *hard* function. Afterwards, we continue with proving certain properties of the function which finally lead to the proof of Theorem 17.

Let n denote the size of the universe, i.e., $n := |U|$. Pick a random subset $S^* \subseteq U$ by sampling each element of U with probability k/n. Thus, the expected size of S^* is k.

Define the function $g : U \mapsto \mathbb{N}$ as $g(S) := |S \cap S^*|$ for any $S \subseteq U$. One can easily verify that g is submodular. We have a positive r whose value will be fixed below. Define the final function $f : U \mapsto \mathbb{N}$ as

$$f(S) := \begin{cases} 1 & \text{if } g(S) = 0 \\ \lceil g(S)/r \rceil & \text{otherwise.} \end{cases}$$

It is not difficult to verify the subadditivity of f; it is also clearly monotone.

In order to prove the core of the hardness result in Lemma 18, we now let $r := \lambda \cdot \frac{mk}{n}$, where $\lambda \geq 1 + \sqrt{\frac{3tn}{mk}}$ and $t = \Omega(\log n)$ will be determined later.

Lemma 18. *An algorithm making at most 2^t queries to the value oracle cannot solve the subadditive maximization problem to within k/r approximation factor.*

Proof. Note that for any $X \subseteq U$, $f(X)$ lies between 0 and $\lceil k/r \rceil$. In fact, the optimal solution is the set S^* whose value is at least k/r. We prove that with high probability the answer to all the queries of the algorithm is one. This implies that the algorithm cannot achieve an approximation ratio better than k/r.

Assume that X_i is the i-th query of the algorithm for $1 \leq i \leq 2^t$. Notice that X_i can be a function of our answers to the previous queries. Define $\mathcal{E}_i$ as the event $f(X_i) = 1$. This is equivalent to $g(X_i) \leq r$. We show that with high probability all events $\mathcal{E}_i$ occur.

For any $1 \leq i \leq 2^t$, we have

$$\Pr\left[\mathcal{E}_i | \bigcap_{j=1}^{i-1} \mathcal{E}_j\right] = \frac{\Pr[\bigcap_{j=1}^{i} \mathcal{E}_j]}{\Pr[\bigcap_{j=1}^{i-1} \mathcal{E}_j]} \geq \Pr\left[\bigcap_{j=1}^{i} \mathcal{E}_j\right] \geq 1 - \sum_{j=1}^{i} \overline{\mathcal{E}_j}.$$

Thus, we have $\Pr[\cap_{i=1}^{2^t} \mathcal{E}_i] \geq 1 - 2^t \sum_{i=1}^{2^t} \Pr[\overline{\mathcal{E}_i}]$ from union bound. Next we bound $\Pr[\mathcal{E}_i]$. Consider a subset $X \subseteq U$ such that $|X| \leq m$. Since the elements of S^* are picked randomly with probability k/n, the expected value of $X \cap S^*$ is at most mk/n. Standard application of Chernoff bounds gives

$$\Pr[f(X) \neq 1] = \Pr[g(X) > r] = \Pr\left[|X \cap S^*| > \lambda \cdot \frac{mk}{n}\right] \leq \exp\Big\{ - (\lambda - 1)^2 \frac{mk}{n} \Big\} \leq \exp\{-3t\} \leq \frac{2^{-2t}}{n},$$

where the last inequality follows from $t \geq \log n$. Therefore, the probability of all $\mathcal{E}_i$ events occurring simultaneously is at least $1 - 1/n$. $\square$

Now we can prove the main theorem of the section.

Proof of Theorem 17. We just need to set $k = m = \sqrt{n}$. Then, $\lambda = \sqrt{3t}$, and the inapproximability ratio is $\Omega(\sqrt{\frac{n}{t}})$. Restricting to polynomial algorithms, we obtain $t := O(\log^{1+\varepsilon} n)$, and considering exponential algorithms with running time $O(2^{t'})$, we have $t = O(t')$, giving the desired results. $\square$

In case the query size is not bounded, we can define $f(X) := 0$ for large sets X, and pull through the same result; however, the function f is no longer monotone in this case.

We now show that the function f is almost submodular. Recall that a function g is submodular if and only if $g(A) + g(B) \geq g(A \cup B) + g(A \cap B)$.

Proposition 19. *For the hard function f defined above, $f(A) + f(B) \geq f(A \cup B) + f(A \cap B) - 2$ always holds; moreover, $f(X)$ is always positive and attains a maximum value of $\tilde{\Theta}(\sqrt{n})$ for the parameters fixed in the proof of Theorem 17.*

5.2 Algorithm

An algorithm that only picks the best item clearly gives a k-competitive ratio. We now show how to achieve an $O(n/k)$ competitive ratio, and thus by combining the two, we obtain an $O(\sqrt{n})$-competitive algorithm for the monotone subadditive secretary problem. This result complements our negative result nicely.

Partition the input stream S into $\ell := n/k$ (almost) equal-sized segments, each of size at most k. Randomly pick all the elements in one of these segments. Let the segments be denoted by $S_1, S_2, \ldots, S_\ell$. Subadditivity of f implies $f(S) \leq \sum_i f(S_i)$. Hence, the expected value of our solution is $\sum_i \frac{1}{\ell} f(S_i) \geq \frac{1}{\ell} f(S) \geq \frac{1}{\ell}\,\mathrm{OPT}$, where the two inequalities follow from subadditivity and monotonicity, respectively.

6 Conclusions and further results

In this paper, we consider the (non-monotone) submodular secretary problem for which we give a constant-competitive algorithm. The result can be generalized when we have a matroid constraint on the set that we pick; in this case we obtain an $O(\log^2 r)$-competitive algorithm where r is the rank of the matroid. However, we show that it is very hard to compete with the optimum if we consider subadditive functions instead of submodular functions. This hardness holds even for "almost submodular" functions; see Proposition 19.

One may consider special non-submodular functions which enjoy certain structural results in order to find better guarantees. For example, let $f(T)$ be the minimum individual value in T which models a bottle-neck situation in the secretary problem, i.e., selecting a group of k secretaries to work together, and the speed (efficiency) of the group is limited to that of the slowest person in the group (note that unlike the submodular case here the condition for employing exactly k secretaries is enforced.) In this case, we present a simple $O(k)$-competitive ratio for the problem as follows. Interview the first $1/k$ fraction of the secretaries without employing anyone. Let α be the highest efficiency among those interviewed. Employ the first k secretaries whose efficiency surpasses α.

Theorem 20. *Following the prescribed approach, we employ the k best secretaries with probability at least $1/e^2k$.*

Indeed we believe that this $O(k)$ competitive ratio for this case should be almost tight. One can verify that provided individual secretary efficiencies are far from each other, say each two consecutive values are farther than a multiplicative factor n, the problem of maximizing the expected value of the minimum efficiency is no easier than being required to employ all the k best secretaries. Theorem 21 in Appendix A provides evidence that the latter problem is hard to approximate.

Another important aggregation function f is that of maximizing the performance of the secretaries we employ: think of picking k candidate secretaries and finally hiring the best. We consider this function in Appendix C for which we present a near-optimal solution. In fact, the problem has been already studied, and an optimal strategy appears in [18]. However, we propose a simpler solution which features certain "robustness" properties (and thus is of its own interest): in particular, suppose we are given a vector $(\gamma_1, \gamma_2, \ldots, \gamma_k)$ such that $\gamma_i \geq \gamma_{i+1}$ for $1 \leq i < k$. Sort the elements in a set R of size k in a non-increasing order, say $a_1, a_2, \ldots, a_k$. The goal is to maximize the efficiency $\sum_i \gamma_i a_i$. The algorithm that we propose maximizes this more general objective obliviously; i.e., the algorithm runs irrespective of the vector γ, however, it can be shown the resulting solution approximates the objective for all vectors γ at the same time. The reader is referred to Appendix C for more details.

Acknowledgments

The second author wishes to thank Bobby Kleinberg for useful discussions.

References

[1] A. A. Ageev and M. I. Sviridenko, *An 0.828-approximation algorithm for the uncapacitated facility location problem*, Discrete Appl. Math., 93 (1999), pp. 149–156.

[2] M. Ajtai, N. Megiddo, and O. Waarts, *Improved algorithms and analysis for secretary problems and generalizations*, SIAM J. Discrete Math., 14 (2001), pp. 1–27.

[3] A. Asadpour, H. Nazerzadeh, and A. Saberi, *Stochastic submodular maximization*, in Proceedings of the 4th International Workshop on Internet and Network Economics (WINE), 2008, pp. 477–489.

[4] M. Babaioff, N. Immorlica, D. Kempe, and R. Kleinberg, *A knapsack secretary problem with applications*, in Proceedings of the 10th International Workshop on Approximation and Combinatorial Optimization (APPROX'07), 2007, pp. 16–28.

[5] ——, *Online auctions and generalized secretary problems*, SIGecom Exch., 7 (2008), pp. 1–11.

[6] M. Babaioff, N. Immorlica, and R. Kleinberg, *Matroids, secretary problems, and online mechanisms*, in Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms (SODA'07), 2007, pp. 434–443.

[7] G. Calinescu, C. Chekuri, M. Pál, and J. Vondrák, *Maximizing a submodular set function subject to a matroid constraint (extended abstract)*, in Integer Programming and Combinatorial Optimization, 12th International IPCO Conference (IPCO'07), 2007, pp. 182–196.

[8] G. Cornuejols, M. Fisher, and G. L. Nemhauser, *On the uncapacitated location problem*, in Studies in integer programming (Proc. Workshop, Bonn. 1975), North-Holland, Amsterdam, 1977, pp. 163–177. Ann. of Discrete Math., Vol. 1.

[9] G. Cornuejols, M. L. Fisher, and G. L. Nemhauser, *Location of bank accounts to optimize float: an analytic study of exact and approximate algorithms*, Management Sci., 23 (1976/77), pp. 789–810.

[10] E. B. Dynkin, *The optimum choice of the instant for stopping a markov process*, Sov. Math. Dokl., 4 (1963), pp. 627–629.

[11] J. Edmonds, *Submodular functions, matroids, and certain polyhedra*, in Combinatorial Structures and their Applications (Proc. Calgary Internat. Conf., Calgary, Alta., 1969), Gordon and Breach, New York, 1970, pp. 69–87.

[12] U. Feige, *A threshold of ln n for approximating set cover*, J. ACM, 45 (1998), pp. 634–652.

[13] U. Feige, *On maximizing welfare when utility functions are subadditive*, in Proceedings of the 38th Annual ACM Symposium on Theory of Computing (STOC'06), ACM, 2006, pp. 41–50.

[14] U. Feige and M. X. Goemans, *Approximating the value of two power proof systems, with applications to max 2sat and max dicut*, in Proceedings of the 3rd Israel Symposium on the Theory of Computing Systems (ISTCS'95), Washington, DC, USA, 1995, IEEE Computer Society, p. 182.

[15] U. Feige, V. S. Mirrokni, and J. Vondrák, *Maximizing non-monotone submodular functions*, in Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07), IEEE Computer Society, 2007, pp. 461–471.

[16] M. L. Fisher, G. L. Nemhauser, and L. A. Wolsey, *An analysis of approximations for maximizing submodular set functions. II*, Math. Programming Stud., (1978), pp. 73–87. Polyhedral combinatorics.

[17] P. R. Freeman, *The secretary problem and its extensions: a review*, Internat. Statist. Rev., 51 (1983), pp. 189–206.

[18] J. P. Gilbert and F. Mosteller, *Recognizing the maximum of a sequence*, J. Amer. Statist. Assoc., 61 (1966), pp. 35–73.

[19] K. S. Glasser, R. Holzsager, and A. Barron, *The d choice secretary problem*, Comm. Statist. C—Sequential Anal., 2 (1983), pp. 177–199.

[20] M. X. Goemans and D. P. Williamson, *Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming*, J. Assoc. Comput. Mach., 42 (1995), pp. 1115–1145.

[21] M. T. Hajiaghayi, R. Kleinberg, and D. C. Parkes, *Adaptive limited-supply online auctions*, in Proceedings of the 5th ACM conference on Electronic Commerce (EC '04), New York, NY, USA, 2004, ACM, pp. 71–80.

[22] M. T. HAJIAGHAYI, R. KLEINBERG, AND T. SANDHOLM, *Automated online mechanism design and prophet inequalities*, in Proceedings of the Twenty-Second National Conference on Artificial Intelligence (AAAI'07), 2007, pp. 58–65.

[23] E. HALPERIN AND U. ZWICK, *Combinatorial approximation algorithms for the maximum directed cut problem*, in Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms (SODA'01), Philadelphia, PA, USA, 2001, Society for Industrial and Applied Mathematics, pp. 1–7.

[24] J. HÅSTAD, *Some optimal inapproximability results*, J. ACM, 48 (2001), pp. 798–859 (electronic).

[25] E. HAZAN, S. SAFRA, AND O. SCHWARTZ, *On the complexity of approximating k-set packing*, Computational Complexity, 15 (2006), pp. 20–39.

[26] N. IMMORLICA, R. D. KLEINBERG, AND M. MAHDIAN, *Secretary problems with competing employers.*, in Proceedings of the 2nd Workshop on Internet and Network Economics (WINE'06), vol. 4286, Springer, 2006, pp. 389–400.

[27] S. IWATA, L. FLEISCHER, AND S. FUJISHIGE, *A combinatorial strongly polynomial algorithm for minimizing submodular functions*, J. ACM, 48 (2001), pp. 761–777 (electronic).

[28] S. KHOT, G. KINDLER, E. MOSSEL, AND R. O'DONNELL, *Optimal inapproximability results for max-cut and other 2-variable csps?*, in Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science (FOCS'04), Washington, DC, USA, 2004, IEEE Computer Society, pp. 146–154.

[29] S. KHULLER, A. MOSS, AND J. NAOR, *The budgeted maximum coverage problem*, Inf. Process. Lett., 70 (1999), pp. 39–45.

[30] R. KLEINBERG, *A multiple-choice secretary algorithm with applications to online auctions*, in Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete Algorithms (SODA '05), Philadelphia, PA, USA, 2005, Society for Industrial and Applied Mathematics, pp. 630–631.

[31] J. LEE, V. MIRROKNI, V. NAGARAJAN, AND M. SVIRIDENKO, *Maximizing non-monotone submodular functions under matroid and knapsack constraints*, in Proceedings of the 41st Annual ACM Symposium on Theory of Computing (STOC), ACM, 2009, pp. 323–332.

[32] L. LOVÁSZ, *Submodular functions and convexity*, in Mathematical programming: the state of the art (Bonn, 1982), Springer, Berlin, 1983, pp. 235–257.

[33] G. L. NEMHAUSER, L. A. WOLSEY, AND M. L. FISHER, *An analysis of approximations for maximizing submodular set functions. I*, Math. Programming, 14 (1978), pp. 265–294.

[34] M. QUEYRANNE, *A combinatorial algorithm for minimizing symmetric submodular functions*, in Proceedings of the sixth annual ACM-SIAM symposium on Discrete algorithms (SODA'95), Philadelphia, PA, USA, 1995, Society for Industrial and Applied Mathematics, pp. 98–101.

[35] A. SCHRIJVER, *A combinatorial algorithm minimizing submodular functions in strongly polynomial time*, J. Combin. Theory Ser. B, 80 (2000), pp. 346–355.

[36] M. SVIRIDENKO, *A note on maximizing a submodular set function subject to a knapsack constraint*, Oper. Res. Lett., 32 (2004), pp. 41–43.

[37] R. J. VANDERBEI, *The optimal choice of a subset of a population*, Math. Oper. Res., 5 (1980), pp. 481–486.

[38] J. VONDRÁK, *Symmetry and approximability of submodular maximization problems*, in Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science (FOCS'09), IEEE Computer Society, 2009. to appear.

[39] J. G. WILSON, *Optimal choice and assignment of the best m of n randomly arriving items*, Stochastic Process. Appl., 39 (1991), pp. 325–343.

A Omitted proofs and theorems

Proof of Lemma 5. Let $k := |B| - |A|$. Then, define in an arbitrary manner sets $\{B_i\}_{i=0}^{k}$ such that

- $B_0 = A$,
- $|B_i \setminus B_{i-1}| = 1$ for $i : 1 \leq i \leq k$,
- and $B_k = B$.

Let $b_i := B_i \setminus B_{i-1}$ for $i : 1 \leq i \leq k$. We can write $f(B) - f(A)$ as follows

$$\begin{aligned} f(B) - f(A) &= \sum_{i=1}^{k} \left[f(B_i) - f(B_{i-1}) \right] \\ &= \sum_{i=1}^{k} \left[f(B_{i-1} \cup \{b_i\}) - f(B_{i-1}) \right] \\ &\leq \sum_{i=1}^{k} \left[f(A \cup b_i) - f(A) \right], \end{aligned}$$

where the last inequality follows from the non-increasing marginal profit property of submodular functions. Noticing that $b_i \in B \setminus A$ and they are distinct, namely $b_i \neq b_{i'}$ for $1 \leq i \neq i' \leq k$, finishes the argument. $\square$

Proof of Lemma 11. The statement follows from the submodularity property, observing that $(R \cup Z) \cap (R \cup Z') = R$, and $f([R \cup Z] \cup [R \cup Z']) \geq 0$. $\square$

Proof of Proposition 19. The function $h(X) := g(X)/r$ is clearly submodular, and we have $h(X) \leq f(X) \leq h(X) + 1$. We obtain $f(A) + f(B) \geq h(A) + h(B) \geq h(A \cup B) + h(A \cap B) \geq f(A \cup B) + f(A \cap B) - 2$. $\square$

Proof of Theorem 20. Let $R = \{a_1, a_2, \ldots, a_{|R|}\} \subseteq S$ denote the set of k best secretaries. Let S^* denote the first $1/k$ fraction of the stream of secretaries. Let $\mathcal{E}^1$ denote the event when $S^* \cap R = \emptyset$, that is, we do not lose the chance of employing the best secretaries (R) by being a mere observer in S^*. Let $\mathcal{E}^2$ denote the event that we finally pick the set R. Let us first bound $\Pr[\mathcal{E}^1]$. In order to do so, define $\mathcal{E}_j^1$ for $j : 1 \leq j \leq |R|$ as the event that $a_j \notin S_e$. We know that $\Pr[\mathcal{E}_1^1] \geq 1/k$. In general, we have for $j > 1$

$$\begin{aligned} \Pr\left[\mathcal{E}_j^1 \middle| \bigcap_{i<j} \mathcal{E}_i^1\right] &\geq \frac{n - \frac{n}{k} - j + 1}{n - j + 1} \\ &\geq \frac{n - \frac{n}{k} - k}{n - k} \\ &= 1 - \frac{n/k}{n-k} \\ &\geq 1 - \frac{2}{k} \qquad \text{assuming } k \leq \frac{n}{2}. \end{aligned} \tag{4}$$

Notice that the final assumption is justified because we can solve the problem of finding the $k' = n - k \leq n/2$ smallest numbers in case $k > n/2$. Using Equation (4) we obtain

$$\begin{aligned} \Pr[\mathcal{E}^1] &= \Pr[\mathcal{E}_1^1] \Pr[\mathcal{E}_2^1 | \mathcal{E}_1^1] \cdots \Pr[\mathcal{E}_{|R|}^1 | \cup_{j<|R|} \mathcal{E}_j^1] \\ &\geq \left(1 - \frac{2}{k}\right)^k \\ &\geq e^{-2}. \end{aligned} \tag{5}$$

The event $\mathcal{E}^2$ happens when $\mathcal{E}^1$ happens and the $(k+1)$th largest element appears in S^*. Thus, we have $\Pr[\mathcal{E}^2] = \Pr[\mathcal{E}^1] \Pr[\mathcal{E}^2 | \mathcal{E}^1] \geq e^{-2} \cdot 1/k = \frac{1}{e^2 k}$. $\square$

Theorem 21. *Any algorithm with a single threshold—i.e., interviewing applicants until some point (observation phase), and then employing any one who is better than all those in the observation phase—misses one of the k best secretaries with probability* $1 - O(\log k/k)$.

Proof. We assume that we cannot find the actual efficiency of a secretary, but we only have an oracle that given two secretaries already interviewed, reports the better of the two. This model is justified if the range of efficiency values is large, and a suitable perturbation is introduced into the values.

Suppose the first secretaries is hired after interviewing a β fraction of the secretaries. If $\beta > \log k/k$ then the probability that we miss at least one of the k best secretaries is at least $1 - (1-\beta)^k = 1 - 1/k$. If on the other hand, β is small, say $\beta \leq \log k/k$, there is little chance that the right threshold can be picked. Notice that in the oracle model, the threshold has to be the efficiency of one prior secretary. Thus for the right threshold to be selected, we need to have the $(k+1)$th best secretary in the first β fraction—the probability of this even is no more than β. Therefore, the probability of success cannot be more than $\log k/k$. □

B Omitted algorithm

Algorithm 3 Monotone Submodular Secretary Algorithm with Matroid constraint

Input: A monotone submodular function $f : 2^{\mathcal{U}} \mapsto \mathbb{R}$, a matroid $(\mathcal{U}, \mathcal{I})$, and a randomly permuted stream of secretaries, denoted by $(a_1, a_2, \ldots, a_n)$.
Output: A subset of secretaries that are independent according to $\mathcal{I}$.

Let $U_1 := \{a_1, a_2, \ldots, a_{\lfloor n/2 \rfloor}\}$
Pick the parameter $k := |S^*|$ uniformly at random from the pool $\{2^0, 2^1, 2^{\log r}\}$
if $k = O(\log r)$ **then**
  Select the best item of the U_1 and output the singleton
else {run Algorithm 1 on U_1 and respect the matroid}
  Let $T_0 \leftarrow \emptyset$
  Let $l \leftarrow \lfloor n/k \rfloor$
  for $i \leftarrow 1$ **to** k **do** {phase i}
    Let $u_i \leftarrow (i-1)l + l/e$
    Let $\alpha_i \leftarrow \max\limits_{\substack{(i-1)l \leq j < u_i \\ T_{i-1} \cup \{a_j\} \in \mathcal{I}}} f(T_{i-1} \cup \{a_j\})$
    if $\alpha_i < f(T_{i-1})$ **then**
      $\alpha_i \leftarrow f(T_{i-1})$
    end if
    Pick an index $p_i : u_i \leq p_i < il$ such that $f(T_{i-1} \cup \{a_{p_i}\}) \geq \alpha_i$ and $T_{i-1} \cup \{a_{p_i}\} \in \mathcal{I}$
    if such an index p_i exists **then**
      Let $T_i \leftarrow T_{i-1} \cup \{a_{p_i}\}$
    else
      Let $T_i \leftarrow T_{i-1}$
    end if
  end for
  Output T_k as the solution
end if

C The secretary problem with the "maximum" function

We now turn to consider a different efficiency aggregation function, namely the maximum of the efficiency of the individual secretaries. Alternately, one can think of this function as a secretary function *with k choices*, that is, we select k secretaries and we are satisfied as long as one of them is the best secretary interviewed. We propose an algorithm that accomplishes this task with probability $1 - O\left(\frac{\ln k}{k}\right)$ for $k > 1$.

As we did before, we assume that n is a multiple of k, and we partition the input stream into k equally-sized segments, named $S_1, S_2, \ldots, S_k$. Let $f(s)$ denote the efficiency of the secretary $s \in S$. For each set $i : 1 \leq i < k$, we compute

$$\alpha_i := \max_{s \in \bigcup_{j \le i} S_i} f(s),$$

which is the efficiency of the best secretary in the first i segments. Clearly, α_i can be computed in an online manner after interviewing the first i segments. For each $i : 1 \le i < k$, we try to employ the first secretary in $\bigcup_{j>i} S_j$ whose efficiency surpasses α_i. Let this choice, if at all present, be denoted s_i. The output of the algorithm consists of all such secretaries $\{s_i\}_i$. Notice that such an element may not exist for a particular i, or we may have $s_i = s_{i'}$ for $i \neq i'$. We employ at most $k-1$ secretaries. The following theorem bounds the failure probability of the algorithm.

Theorem 22. *The probability of not employing the best secretary is $O\left(\frac{\ln k}{k}\right)$.*

Proof. Let $(a_1, a_2, \ldots, a_n)$ denote the stream of interviewed secretaries. Let a_m be the best secretary, and suppose $a_m \in S_i$, namely $(i-1)l < m \le il$, where $l := n/k$. Our algorithm is successful if the second best secretary of the set $\{a_1, a_2, \ldots, a_{m-1}\}$ does not belong to S_i. The probability of this event is

$$\frac{(i-1)l}{m} \ge \frac{(i-1)l}{il} = \frac{i-1}{i}. \tag{6}$$

The probability of $a_m \in S_i$ is $1/k$ and conditioned on this event, the probability of failure is at most $1/i$. Hence, the total failure probability is no more than $\sum_{i=1}^{k} \frac{1}{k}\frac{1}{i} = O\left(\frac{\ln k}{k}\right)$ as claimed. □

This problem has been previously studied by Gilbert and Mosteller [18]. Our algorithm above is simpler and yet "robust" in the following sense. The primary goal is to select the best secretary, but we also guarantee that many of the "good" secretaries are also selected. In particular, we show that the better the rank of a secretary is in our evaluation, the higher is the guarantee we have for employing her.

Theorem 23. *The probability of not hiring a secretary of rank y is $O\left(\sqrt{\frac{y}{k}}\right)$.*

Proof. Let $(a_1, a_2, \ldots, a_n)$ denote the stream of interviewed secretaries. Let a_m be the secretary of rank y, and suppose $a_m \in S_i$, namely $(i-1)l < m \le il$, where $l := n/k$. Below we define three bad events whose probabilities we bound, and we show that a_m is hired provided none of these events occur. In particular, we give an upper bound of $O(\sqrt{y/k}$ for each event. The claim then follows from the union bound.

Let $z := \sqrt{\frac{k}{y-1}} - 1$. The event $\mathcal{E}_1$ occurs if $i \le z$. This event happens with probability z/k which is less than $\sqrt{\frac{1}{k(y-1)}} \le \sqrt{\frac{y}{k}}$.

We say the event $\mathcal{E}_2$ happens if a_m is not the best secretary among those in sets $S_i, S_{i-1}, \ldots, S_{i-z}$. This happens when there is at least one of the $y-1$ secretaries better than a_m in these sets. Let W be a random variable denoting the number of these $y-1$ secretaries in any of the mentioned sets. Since any secretary is in one of these sets with probability $(z+1)/k$ (notice that $z+1$ is the number of these sets), we can say that the expected value of W is $(y-1)(z+1)/k$. Using the Markov Inequality, the probability that W is at least 1 is at most its expected value which is $(y-1)(z+1)/k$. Thus, using the definition of z, we get an upper bound of $O\left(\sqrt{\frac{y-1}{k}}\right)$ for $\mathcal{E}_2$.

Finally, we define $\mathcal{E}_3$ as the event when the best secretary among $\{a_{(i-z-1)l+1}, a_{(i-z-1)l+2}, \ldots, a_{j-1}\}$ (secretaries appearing before a_m in the above-mentioned sets) is in set S_i. This happens with probability at most $1/(z+1)$, because there are $z+1$ sets that the best secretary is equally likely to be in each. Thus, we get $\Pr[\mathcal{E}_3] = O\left(\sqrt{\frac{y}{k}}\right)$ by definition of z.

If non of the events $\mathcal{E}_1, \mathcal{E}_2$ and $\mathcal{E}_3$ happen, we claim a_m is employed. Because if the maximum item of items $\{a_{(i-z-1)l+1}, a_{(i-z-1)l+2}, \ldots, a_{j-1}\}$ is in the set $S_{i'}$, and $i - z \le i' < i$, then we hire a_m for the set $S_{i'}$; refer to the algorithm when we consider the threshold $\alpha_{i'}$. □

The aforementioned algorithm of [18] misses a *good* secretary of rank y with probability roughly $1/y$. On the other hand, one can show that the algorithm of Kleinberg [30] (for maximizing the sum of the efficiencies of the secretaries) picks secretaries of high rank with probability about $1 - \Theta(1/\sqrt{k})$. However, the latter algorithm guarantees the selection of the best secretary with a probability no more than $O(1/\sqrt{k})$. Therefore, our algorithm has the nice features of both these algorithms: the best secretary is hired with a very good probability, while other good secretaries also have a good chance of being employed.