Improved Competitive Ratios for Submodular Secretary Problems

(Extended Abstract)

Moran Feldman*, Joseph (Seffi) Naor**, and Roy Schwartz

Computer Science Dept., Technion, Haifa, Israel
{moranfe,naor,schwartz}@cs.technion.ac.il

Abstract. The Classical Secretary Problem was introduced during the 60's of the 20^{th} century, nobody is sure exactly when. Since its introduction, many variants of the problem have been proposed and researched. In the classical secretary problem, and many of its variant, the input (which is a set of secretaries, or elements) arrives in a random order. In this paper we apply to the secretary problem a simple observation which states that the random order of the input can be generated by independently choosing a random continuous arrival time for each secretary. Surprisingly, this simple observation enables us to improve the competitive ratio of several known and studied variants of the secretary problem. In addition, in some cases the proofs we provide assuming random arrival times are shorter and simpler in comparison to existing proofs. In this work we consider three variants of the secretary problem, all of which have the same objective of maximizing the value of the chosen set of secretaries given a monotone submodular function f. In the first variant we are allowed to hire a set of secretaries only if it is an independent set of a given partition matroid. The second variant allows us to choose any set of up to k secretaries. In the last and third variant, we can hire any set of secretaries satisfying a given knapsack constraint.

1 Introduction

In the (classical) secretary problem (CS), a set of n secretaries arrives in a random order for an interview. Each secretary is associated with a distinct non-negative value which is revealed upon arrival, and the objective of the interviewer is to choose the best secretary (the one having maximum value). The interviewer must decide after the interview whether to choose the candidate or not. This decision is irrevocable and cannot be altered later. The goal is to maximize the probability of choosing the best secretary.[1] It is known that the optimal algorithm for CS

* Moran Feldman is a recipient of the Google Europe Fellowship in Market Algorithms, and this research is supported in part by this Google Fellowship.
** This work was partly supported by ISF grant 1366/07.
[1] The above definition of CS is slightly different from the standard one, though both definitions are equivalent. In the standard definition, the secretaries have ranks instead of values, and the relative rank of each secretary is revealed upon arrival.

is to reject the first n/e secretaries, and then choose the first secretary that is better than any of the first n/e secretaries. This algorithm succeeds in finding the best secretary with probability of e^{-1} [10, 24].

Throughout the years, many variants of CS have been considered. In this work we focus on variants where a subset of the secretaries can be chosen and the goal is to maximize some function of this chosen subset. Such variants of CS have attracted much attention (see, e.g., [3–5, 17, 19]). Some examples of these variants include: requiring the subset of chosen secretaries to form an independent set in a given matroid, limiting the size of the subset to at most k, and requiring the chosen secretaries to satisfy some knapsack constraints. All these variants have been studied with both linear and submodular objectives. More details on previous results can be found in Section 1.2.

In this work we use a simple observation which has been employed in problems other than CS. The observation states that the random order in which the secretaries arrive can be modeled by assigning each secretary an independent uniform random variable in the range $[0, 1)$. This continuous random variable determines the time in which the secretary arrives. Obviously, this modeling is equivalent to a random arrival order.[2] Though this modeling of the arrival times as continuous random variables is very simple, it has several advantages when applied to variants of the secretary problem, on which we elaborate now. First, it enables us to achieve better competitive ratios for several variants of CS. Second, the proofs of the performance guarantees of the algorithms are much simpler. The latter can be exemplified by the following simple problem.

Assume one wishes to partition the arriving secretaries into two sets where each secretary independently and uniformly chooses one of the sets, and all secretaries of one set arrive before all secretaries of the other set. An obvious difficulty is that the position of a secretary in the random arrival order depends on the positions of all other secretaries. For example, if a set S contains many secretaries that have arrived early, then a secretary outside of S is likely to arrive late, since many of the early positions are already taken by members of S. This difficulty complicates both the algorithms and their analysis. To get around this dependence [3, 19], for example, partition the secretaries into two sets: one containing the first m secretaries and the other containing the last $n-m$ secretaries. The value of m is binomially distributed $\mathsf{Bin}(n, 1/2)$. It can be shown that this partition, together with the randomness of the input, guarantees that every secretary is uniformly and independently assigned to one of the two sets. Such an elaborate argument is needed to create the desired partitioning because of the dependencies between positions of secretaries in the arrival order.

In contrast, using the modeling of the arrival times as continuous random variables, creating a subset of secretaries where each secretary independently belongs to it with probability $1/2$ is simple. One just has to choose all secretaries that arrive before time $t = 1/2$. This simplifies the above argument, designed for a random arrival order, considerably. This simple example shows how continuous

[2] Given a random arrival order, we can sample n independent uniformly random arrival times, sort them and assign them sequentially to the secretaries upon arrival.

arrival times can be used to simplify both the algorithm and its analysis. In the variants of CS considered in this paper, this simplification enables us to obtain improved competitive ratios.

For some variants, the algorithms presented in this paper can be viewed as the continuous "counterparts" of known algorithms (see the uniform matroid and knapsack variants), but this is not always the case. For the partition matroid variant, the algorithm we define in this work using continuous arrival times uses different techniques than previous known algorithms for this case.

It is important to note that the continuous time "counterparts" of algorithms designed using a random arrival order are *not* equivalent to the original algorithms. For example, the optimal algorithm for CS assuming continuous random arrival times is to inspect secretaries up to time e^{-1}, and hire the first secretary after this time which is better than any previously seen secretary (see Appendix A for an analysis of this algorithm using continuous arrival times). Observe that this algorithm inspects n/e secretaries in *expectation*, while the classical algorithm inspects that number of secretaries *exactly*. This subtle difference is what enables us to improve and simplify previous results.

Formally, all problems considered in this paper are online problems in which the input consists of a set of secretaries (elements) arriving in a random order. Consider an algorithm $\mathcal{A}$ for such a problem, and denote by opt an optimal offline algorithm for the problem. Let I be an instance of the problem, and let $\mathcal{A}(I)$ and $\mathsf{opt}(I)$ be the values of the outputs of $\mathcal{A}$ and opt, respectively, given I. We say that A is α-competitive (or has an α-competitive ratio) if $\inf_I \frac{\mathbb{E}[\mathcal{A}(I)]}{\mathsf{opt}(I)} \geq \alpha$, where the expectation is over the random arrival order of the secretaries of I and the randomness of $\mathcal{A}$ (unless $\mathcal{A}$ is deterministic). The competitive ratio is a standard measure for the quality of an online algorithm.

1.1 Our Results

In this paper we consider variants of CS where the objective function is normalized, monotone and submodular.[3] There are three variants for which we provide improved competitive ratios. The first is the *submodular partition matroid secretary* problem (SPMS) in which the secretaries are partitioned into subsets, and at most one secretary from each subset can be chosen. The second is the *submodular cardinality secretary* problem (SCS) in which up to k secretaries can be chosen. The third and last variant is the *submodular knapsack secretary* problem (SKS), in which each secretary also has a cost (which is revealed upon arrival), and any subset of secretaries is feasible as long as the total cost of the subset does not exceed a given budget.

For SPMS we present a competitive ratio of $(1 - \ln 2)/2 \approx 0.153$, which improves on the current best result of $\Omega(1)$ by Gupta et al. [17]. We note that the exact competitive ratio given by [17] is not stated explicitly, however, by

[3] Given a groundset $\mathcal{S}$, a function $f : 2^{\mathcal{S}} \to \mathbb{R}$ is called *submodular* if for every $A, B \subseteq \mathcal{S}$, $f(A) + f(B) \geq f(A \cup B) + f(A \cap B)$. Additionally, f is called *normalized* if $f(\varnothing) = 0$ and *monotone* if for every $A \subseteq B \subseteq \mathcal{S}$, $f(A) \leq f(B)$.

inspecting their result carefully it seems that the competitive ratio they achieve is at most 71/1280000. We note that for SPMS the algorithm we provide is not a continuous time "counterpart" of the algorithm of [17]. This demonstrates the fact that modeling the arrival times as continuous random variables helps in designing and analyzing algorithms for submodular variants of CS.

For SCS we present a competitive ratio of $(e-1)/(e^2+e) \approx 0.170$, and the current best result for this problem is due to Bateni et al. [5] who provided a $(1-e^{-1})/7 \approx 0.0903$ competitive ratio. There are two points to notice when comparing our result and that of [5]. First, [5] did not optimize the competitive ratio analysis of their algorithm. In fact, their algorithm provides better ratios then stated, however, it does not seem that their competitive ratio reaches 0.17. Hence, in this paper we still obtain improved competitive ratios, though the improvement is smaller than stated above. Second, the algorithm presented in this paper for SCS can be seen as a continuous time "counterpart" of [5]'s algorithm. However, our analysis is simpler than the analysis presented in [5], and also enables us to provide improved competitive ratios.

For SKS we provide a competitive ratio of $(20e)^{-1} \approx 0.0184$. The current best result is due to Bateni et al. [5] which provide a ratio of $\Omega(1)$. The exact competitive ratio is not stated in [5], but careful inspection of their algorithm shows that it is at most $96^{-1} \approx 0.0104$. Notice that the best known competitive ratio for the *linear* version of SKS is only $10e^{-1} \approx 0.0368$ [3]. As before, the algorithm presented in this paper for SKS can be seen as a continuous time "counterpart" of [5]'s algorithm. However, our analysis is simpler than the analysis presented in [5], enabling us to provide improved competitive ratios. Table 1 summarizes the above results.

1.2 Related Work

Many variants of CS have been considered throughout the years and we shall mention here only those most relevant to this work. Babaioff et al. [4] considered the case where the chosen subset of secretaries needs to be an independent set of a given matroid, and the objective function f is linear. They provided a competitive ratio of $\Omega(\log^{-1} r)$ for this problem, where r is the rank of the matroid. For several specific matroids, better constant competitive ratios are

Table 1. Comparison of our results with the known results for the monotone submodular and linear variants of the problems we consider.

Problem	Our Result	Previous Result	Best Result for Linear Variant
SPMS	0.153	0.0000555 [17]	0.368[1]
SCS	0.170	0.0903 [5]	0.368 [3]
SKS	0.0184	0.0104 [5]	0.0368 [3]

[1] For linear objective functions one can apply the algorithm for the classical secretary problem to each subset of secretaries independently.

known $[4, 9, 18, 20]$. The special variant of SCS where the objective function is linear has also been studied. Two incomparable competitive ratios were obtained by Babaioff et al. [3] and Kleinberg [19], achieving competitive ratios of e^{-1} and $1 - O(1/\sqrt{k})$, respectively. An interesting variant of SCS with a linear objective function gives each of the k possible slots of secretaries a different weight. The value of the objective function in this case is the sum of the products of a slots' weights with the values of the secretaries assigned to them. Babaioff et al. [2] provide a competitive ratio of $1/4$ for this special variant. Additional variants of CS can found in $[1, 6, 13, 14, 16]$.

Another rich field of study is that of submodular optimization, namely optimization problems in which the objective function is submodular. In recent years many new results in this field have been achieved. The most basic problem, in this field, is that of unconstrained maximization of a nonmonotone submodular function $[12, 15]$. Other works consider the maximization of a nonmonotone submodular function under various combinatorial constraints $[17, 22, 25]$. The maximization of a monotone submodular function under various combinatorial constraints (such as a matroid, the intersection of several matroids and knapsack constraints) has also been widely studied $[7, 8, 21, 23, 25]$.

Thus, it comes as no surprise that recent works have combined the secretary problem with submodular optimization. Gupta et al. [17] were the first to consider this combination. For the variant where the goal is to maximize a submodular function of the chosen subset of secretaries under the constraint that this subset is independent in a given matroid, Gupta et al. [17] provide a competitive ratio of $\Omega(\log^{-1} r)$ (where r is the rank of the matroid). If the constraint is that the chosen subset of secretaries belongs to the intersection of ℓ matroids, Bateni et al. [5] provide a competitive ratio of $\Omega(\ell^{-1} \log^{-2} r)$. If the objective function is submodular and *monotone*, the special case of a partition matroid is exactly SPMS, and the special case of a uniform matroid is exactly SCS. As mentioned before, for SPMS, Gupta et al. [17] provide a competitive ratio of $\Omega(1)$ for SPMS which is at most $71/1280000$ (the exact constant is not explicitly stated in their work). They also get a similar competitive ratio for a variant of SPMS with a non-monotone submodular objective function. For SCS, Gupta et al. [17] provide a competitive ratio of $\Omega(1)$ which is at most $1/1417$ (again, the exact constant is not explicitly stated in their work), and a bit more complex $\Omega(1)$ ratio for a variant of SCS with a non-monotone submodular objective function. Both ratios were improved by Bateni et al. [5] who provided a competitive ratio of $(1 - e^{-1})/7 \approx 0.0903$ for SCS, and a $e^{-2}/8 \approx 0.0169$-competitive algorithm for its non-monotone variant. For SKS the current best result is due to Bateni et al. [5] who provide a competitive ratio of at most $96^{-1} \approx 0.0104$ (as before, the exact constant is not explicitly stated in their work). Another extension considered by [5] is a generalization of SKS where every secretary has a ℓ dimensional cost, and the total cost of the secretaries in each dimension should not exceed the budget of this dimension (i.e., the hired secretaries should obey ℓ knapsack constraints). For this problem [5] gives an $\Omega(\ell^{-1})$ competitive algorithm.

Organization. Section 2 contains formal definitions and several technical lemmata. Sections 3, 4 and 5 provide the improved competitive ratio for SPMS, SCS and SKS, respectively.

2 Preliminaries

An instance of a constrained secretary problem consists of three components $\mathcal{S}, \mathcal{I}$ and f.

- $\mathcal{S}$ is a set of n secretaries arriving in a random order.[4]
- $\mathcal{I} \subseteq 2^{\mathcal{S}}$ is a collection of independent sets of secretaries. The sets in $\mathcal{I}$ are known in advance in some settings (e.g., SCS), and are revealed over time in other settings (e.g., SKS). However, at any given time, we know all independent sets containing only secretaries that already arrived.
- $f : 2^{\mathcal{S}} \to \mathbb{R}$ is a function over the set of secretaries accessed using an oracle which given a set S of secretaries that have already arrived, returns $f(S)$.

The goal is to maintain an independent set R of secretaries, and maximize the final value of $f(R)$ (i.e., its value after all secretaries have arrived). Upon arrival of a secretary s, the algorithm has to either add it to R (assuming $R \cup \{s\} \in \mathcal{I}$), or reject it. Either way, the decision is irrevocable. Given a submodular function $f : \mathcal{S} \to \mathbb{R}^+$, the discrete derivative of f with respect to s is $f_s(R) = f(R \cup \{s\}) - f(R)$. We use this shorthand throughout the paper.

Most algorithms for secretary problems with linear objective functions require every secretary to have a distinct value. This requirement does not make sense for submodular objective functions, and therefore, we work around it by introducing a total order over the secretaries, which is a standard practice (see, e.g., [9]). Formally, we assume the existence of an arbitrary fixed order Z over the secretaries. If such an order does not exist, it can be mimicked by starting with an empty ordering, and placing every secretary at a random place in this ordering upon arrival. The resulting order is independent of the arrival order of the secretaries, and therefore, can be used instead of a fixed order. Let s_1, s_2 be two secretaries, and let S be a set of secretaries. Using order Z we define $s_1 \succ_S s_2$ to denote "$f_{s_1}(S) > f_{s_2}(S)$, or $f_{s_1}(S) = f_{s_2}(S)$ and s_1 precedes s_2 in Z". Notice that $\succ_S$ is defined using f and Z. Whenever we use $\succ_S$, we assume f is understood from context and Z is the order defined above.
Remark: The probability that two secretaries arrive at the same time is 0, thus we ignore this event.

The following theorem is used occasionally in our proofs. Similar theorems appear in [12, 11].

---

[4] If the input is given as a random permutation, the size n of $\mathcal{S}$ is assumed to be known. Note that in order to generate the random arrival times of the secretaries and assign them upon arrival, n has to be known in advance. On the other hand, if the arrival times are part of the input, the algorithms in this work need not know n.

Theorem 1. *Given a normalized monotone submodular function $f : 2^{\mathcal{S}} \to \mathbb{R}^+$, a set A and a random set A' containing every element of A with probability at least p (not necessarily independently). Then, $\mathbb{E}[f(A')] \geq p \cdot f(A)$.*

Proof. Order the elements of A in an arbitrary order $\{a_1, a_2, \ldots, a_{|A|}\}$. Let X_i be an indicator for the event $\{a_i \in A'\}$, and let $A_i = \{a_1, a_2, \ldots, a_i\}$. Then,

$$\mathbb{E}[f(A')] = \mathbb{E}\left[\sum_{i=1}^{|A|} X_i \cdot f_{a_i}(A' \cap A_{i-1})\right] =$$

$$\sum_{i=1}^{|A|} \Pr[X_i = 1] \cdot \mathbb{E}\left[f_{a_i}(A' \cap A_{i-1}) \,|X_i = 1\right] \geq \sum_{i=1}^{|A|} p \cdot f_{a_i}(A_{i-1}) = p \cdot f(A) \enspace ,$$

where the inequality follows from the submodularity of f.

3 $(1 - \ln 2)/2 \approx 0.153$-Competitive Algorithm for SPMS

The Submodular Partition Matroid Secretary Problem (SPMS) is a secretary problem with a normalized monotone and submodular objective function f. The collection $\mathcal{I}$ of independent sets is determined by $G_1 \times \ldots \times G_k$ (where the G_i's are a partition of $\mathcal{S}$). This variant corresponds to the scenario where the goal is to hire secretaries of different types, one of each type. For every secretary s, the index of the set G_i containing s is revealed when s arrives.

When designing an algorithm for SPMS, we want to select the best secretary from every set G_i. If f was linear, we could apply the algorithm for the classical secretary problem to each G_i separately. The following algorithm is based on a similar idea.

SPMS Algorithm(f, k)**:**

1. Initialize $R \leftarrow \varnothing$.
2. Observe the secretaries arriving till time t.[a]
3. After time t, for every secretary s arriving, let G_i be the set of s. Accept s into R if:
   (a) no previous secretary of G_i was accepted,
   (b) and for every previously seen $s' \in G_i$, $s' \prec_R s$.
4. Return R.

[a] t is a constant to be determined later.

In this subsection we prove the following theorem.

Theorem 2. *The above algorithm is a $(1 - \ln 2)/2 \approx 0.153$-competitive algorithm for SPMS.*

The algorithm clearly maintains R as a feasible set of secretaries, hence, we only need to show that, in expectation, it finds a good set of secretaries.

Observation 3 *We can assume there is at least one secretary in every set G_i.*

Proof. The behavior of the algorithm is not effected by empty sets G_i.

3.1 Analysis of a single G_i

In this subsection we focus on a single G_i. Let E_i be an event consisting of the arrival times of all secretaries in $\mathcal{S} - G_i$, we assume throughout this subsection that some fixed event E_i occurred. Let R_x be the set of secretaries from $\mathcal{S} - G_i$ collected by the algorithm up to time x assuming no secretary of G_i arrives (observe that R_x is not random because we fixed E_i). We define $\hat{s}_x$ as the maximum secretary in G_i with respect to $\succ_{R_x}$. The analysis requires two additional event types: A_x is the event that $\hat{s}_x$ arrives at time x, and B_x is the same event with the additional requirement that the algorithm collected $\hat{s}_x$.

Lemma 1. *For every $x > t$, $\Pr[B_x|A_x] \geq 1 - \ln x + \ln t$.*

Proof. Event A_x states that $\hat{s}_x$ arrived at time x. If no other secretary of G_i is collected till time x, $\hat{s}_x$ is collected by the definition of the algorithm. Hence, it is enough to bound the probability that no secretary of G_i is collected till time x, given A_x.

Observe that R_x takes at most $k-1$ values for $x \in [0, 1)$. Hence, the range $[0, 1)$ can be divided into k intervals $\mathcal{I}_1, \ldots, \mathcal{I}_k$ such that the set R_x is identical for all times within one interval. Divide the range $[0, x)$ into small steps of size Δy such that Δy divides t and x, and every step is entirely included in an interval (this is guaranteed to happen if Δy also divides the start time and the length of every interval $\mathcal{I}_i$). Since each step is entirely included in a single interval, for every time x in step j, $R_x = R_{(j-1)\cdot\Delta y}$.

A secretary cannot be collected in step j if $j \cdot \Delta y \leq t$. If this is not the case, a secretary is collected in step j if the maximum secretary of G_i in the range $[0, j \cdot \Delta y)$ with respect to $\succ_{R_{(j-1)\cdot\Delta y}}$ arrives at time $(j-1) \cdot \Delta y$ or later. The probability that this happens is $\Delta y/(j \cdot \Delta y) = j^{-1}$. We can now use the union bound to upper bound the probability that any secretary is accepted in any of the steps before time x:

$$\Pr[B_x|A_x] \geq 1 - \sum_{j=t/\Delta y+1}^{x/\Delta y} j^{-1} \geq 1 - \int_{t/\Delta y}^{x/\Delta y} \frac{dj}{j} = 1 - [\ln j]_{t/\Delta y}^{x/\Delta y} = 1 - \ln x + \ln t \ .$$

Let s_i^* denote the single secretary of $G_i \cap OPT$, and let a_i be the secretary of G_i collected by the algorithm. If no secretary is collected from G_i, assume a_i is a dummy secretary of value 0 (i.e., f is oblivious to the existence of this dummy secretary in a set). We also define R_i to be the set R immediately before the algorithm collects a_i (if the algorithms collects no secretary of G_i, R_i is an arbitrary set).

Observation 4 *If B_x occurs for some x, $f_{a_i}(R_i) \geq f_{s_i^*}(R)$, where R is the set returned by the algorithm.*

Proof.

$$f_{a_i}(R_i) \overset{(1)}{=} f_{a_i}(R_x) \overset{(2)}{\geq} f_{s_i^*}(R_x) \overset{(3)}{\geq} f_{s_i^*}(R) \ .$$

Where (1) and (2) follow from the fact that B_x occurred, and therefore, a_i was collected at time x and $a_i = \hat{s}_x$. Also, B_x implies $R_x \subseteq R$, hence, the submodularity of f implies (3).

Let $B_i = \cup_{x \in (t,1)} B_x$, and let P_i be $\{s_i^*\}$ if B_i occurred, and $\varnothing$, otherwise.

Corollary 1. $f_{a_i}(R_i) \geq f(R \cup P_i) - f(R)$.

Proof. If $P_i = \varnothing$, the claim follows because $f_{a_i}(R_i)$ is nonnegative by the monotonicity of f. If $P_i = \{s_i^*\}$, we know that B_i occurred, and therefore, there must be an x for which B_x occurred also. The corollary now follows immediately from Observation 4.

Lemma 2. $\Pr[B_i] \geq 2 + \ln t - 2t$.

Proof. Observe that the events B_x are disjoint, hence, $\Pr[B_i]$ is the sum of the probabilities of the events B_x. Since B_x implies A_x, $\Pr[B_x] = \Pr[B_x | A_x] \cdot \Pr[A_x]$. The event A_x requires that the maximum secretary with respect to $\succ_{R_x}$ arrives in time x. The probability that this secretary arrives in an interval of size Δx is Δx. Hence, the probability that it arrives in an infinitesimal interval of size dx is $\Pr[A_x] = dx$. Therefore,

$$\Pr[B_i] = \int_t^1 \Pr[B_x | A_x] dx \geq \int_t^1 (1 - \ln x + \ln t) dx = 2 + \ln t - 2t \ .$$

The last expression is maximized for $t = 0.5$, thus, we choose $t = 0.5$.

3.2 Analysis of the Entire Output

Throughout the previous subsection we assumed some fixed event E_i occurred. Hence, Corollary 1 and Lemma 2 were proven given this assumption, however, they are also true without it.

Lemma 3. *Corollary 1 and Lemma 2 also hold without fixing an event E_i.*

Proof. Corollary 1 states that for every fixed E_i, if B_i occurs then $f_{a_i}(R_i) \geq f(R \cup P_i) - f(R)$. Since some event E_i must occur (the secretaries of $\mathcal{S} - G_i$ must arrive at some times), this is also true without fixing some E_i.

Let us rephrase Lemma 2 to explicitly present the assumption that some fixed E_i occurred: $\Pr[B_i | E_i] \geq 1 - \ln 2$ (recall that we chose $t = 0.5$). Therefore,

$$\Pr[B_i] = \sum_{E_i} \Pr[E_i] \cdot \Pr[B_i | E_i] \geq (1 - \ln 2) \cdot \sum_{E_i} \Pr[E_i] = 1 - \ln 2 \ .$$

Let $P = \cup_{i=1}^{k} P_i$, and notice that $P \subseteq OPT$. The following lemma lower bounds the expected value of $f(P)$.

Lemma 4. $\mathbb{E}[f(P)] \geq (1 - \ln 2) f(OPT)$.

Proof. Every element $s_i^* \in OPT$ appears in P with probability $\Pr[B_i]$. By Lemma 2 and the value we chose for t, the last probability is at least $1 - \ln 2$. Hence, the lemma follows from Theorem 1.

We are now ready to prove Theorem 2.

Proof (Proof of Theorem 2). Observe the following.

$$\mathbb{E}[f(R)] = \mathbb{E}\left[\sum_{i=1}^{k} f_{a_i}(R_i)\right] \geq \mathbb{E}\left[\sum_{i=1}^{k} f(R \cup P_i) - f(R)\right] \geq \mathbb{E}[f(R \cup P)] - \mathbb{E}[f(R)].$$

Rearranging terms, we get: $\mathbb{E}[f(R)] \geq \frac{f(R \cup P)}{2} \geq \frac{f(P)}{2} \geq \frac{1-\ln 2}{2} \cdot f(OPT)$.

4 The Submodular Cardinality Secretary Problem

The Submodular Cardinality Secretary Problem (SCS) is a secretary problem in which the objective function f is a normalized monotone submodular function, and we are allowed to hire up to k secretaries (the scollection $\mathcal{I}$ of independent sets contains every set of up to k secretaries).

Theorem 5. *There is a $(e-1)/(e^2+e) \approx 0.170$-competitive algorithm for SCS.*

Due to space limitations, the proof of Theorem 5 is omitted from this extended abstract.

5 The Submodular Knapsack Secretary Problem

The Submodular Knapsack Secretary Problem (SKS) is a secretary problem in which the objective function f is a normalized monotone submodular function and every secretary s has a cost $c(s)$ (revealed upon arrival). A budget B is also given as part of the input, and the algorithm is allowed to hire secretaries as long as it does not exceed the budget. In other words, the collection $\mathcal{I}$ of allowed sets contains every set of secretaries whose total cost is at most B.

Theorem 6. *There is a $1/(20e)$-competitive algorithm for SKS.*

Due to space limitations, the proof of Theorem 6 is omitted from this extended abstract.

References

1. Ajtai, M., Megiddo, N., Waarts, O.: Improved algorithms and analysis for secretary problems and generalizations. SIAM J. Discrete Math 14(1), 1–27 (2001)
2. Babaioff, M., Dinitz, M., Gupta, A., Immorlica, N., Talwar, K.: Secretary problems: Weights and discounts. In: 20th ACM-SIAM Symposium on Discrete Algorithms, pp. 1245–1254. Society for Industrial and Applied Mathematics, Philadelphia, PA (2009)

3. Babaioff, M., Immorlica, N., Kempe, D., Kleinberg, R.: A knapsack secretary problem with applications. In: Charikar, M., Jansen, K., Reingold, O., Rolim, J. (eds.) Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques. LNCS, vol. 4627, pp. 16–28. Springer, Heidelberg (2007)
4. Babaioff, M., Immorlica, N., Kleinberg, R.: Matroids, secretary problems, and online mechanisms. In: 18th ACM-SIAM Symposium on Discrete Algorithms, pp. 434–443. Society for Industrial and Applied Mathematics, Philadelphia, PA (2007)
5. Bateni, M.H., Hajiaghayi, M.T., Zadimoghaddam, M.: Submodular secretary problem and extensions. In: Serna, M., Shaltiel, R., Jansen, K., Rolim, J. (eds.) Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques. LNCS, vol. 6302, pp. 39–52. Springer, Heidelberg (2010)
6. Buchbinder, N., Jain, K., Singh, M.: Secretary problems via linear programming. In: Eisenbrand, F., Shepherd, F.B. (eds.) Integer Programming and Combinatorial Optimization. LNCS, vol. 6080, pp. 163–176. Springer, Heidelberg (2010)
7. Calinescu, G., Chekuri, C., Pál, M., Vondrák, J.: Maximizing a monotone submodular function subject to a matroid constraint. To appear in SIAM J. Comput.
8. Chekuri, C., Vondrák, J., Zenklusen, R.: Submodular function maximization via the multilinear relaxation and contention resolution schemes. In: 42nd ACM Symposium on Theory of Computer Science, pp. 783–792. ACM, New York, NY (2011)
9. Dimitrov, N.B., Plaxton, C.G.: Competitive weighted matching in transversal matroids. In: Aceto, L., Damgărd, I., Goldberg, L., Halldórsson, M., Ingólfsdóttir, A., Walukiewicz, I. (eds.) Automata, Languages and Programming. LNCS, vol. 5125, pp. 397–408. Springer, Heidelberg (2008)
10. Dynkin, E.B.: The optimum choice of the instant for stopping a markov process. Sov. Math. Dokl. 4, 627–629 (1963)
11. Feige, U.: On maximizing welfare when utility functions are subadditive. SIAM J. Comput. 39(1), 122–142 (2009)
12. Feige, U., Mirrokni, V.S., Vondrák, J.: Maximizing non-monotone submodular functions. In: 48th Annual IEEE Symposium on Foundations of Computer Science, pp. 461–471. IEEE Computer Society, Washington DC (2007)
13. Ferguson, T.S.: Who solved the secretary problem? Statistical Science 4(3), 282–289 (1989)
14. Freeman, P.R.: The secretary problem and its extensions: A review. International Statistical Review 51(2), 189–206 (1983)
15. Gharan, S.O., Vondrák, J.: Submodular maximization by simulated annealing. In: 22nd ACM-SIAM Symposium on Discrete Algorithms, pp. 1096–1116 (2011)
16. Gilbert, J., Mosteller, F.: Recognizing the maximum of a sequence.: In: Fienberg, S. Hoaglin, D. (eds) Selected Papers of Frederick Mosteller. Springer Series in Statistics, pp. 355–398. Springer, New York (2006)
17. Gupta, A., Roth, A., Schoenebeck, G., Talwar, K.: Constrainted non-monotone submodular maximization: Offline and secretary algorithms. In: Saberi, A. (eds.) Internet and Network Economics. LNCS, vol. 6484, pp. 246–257. Springer, Heidelberg (2010)
18. Im, S. Wang, Y.: Secretary problems: Laminar matroid and interval scheduling. In: 22nd ACM-SIAM Symposium on Discrete Algorithms, pp. 1096–1116 (2011)
19. Kleinberg, R.. A multiple-choice secretary algorithm with applications to online auctions. 16th ACM-SIAM Symposium on Discrete Algorithms, pp. 630–631. Society for Industrial and Applied Mathematics, Philadelphia, PA (2005)
20. Korula, N., Pál, M.: Algorithms for secretary problems on graphs and hypergraphs. In: Albers, S., Marchetti-Spaccamela, A., Matias, Y., Nikoletseas, S., Thomas, W.

(eds.) Automata, Languages and Programming. LNCS, vol. 5556, pp. 508–520. Springer, Heidelberg (2009)
21. Kulik, A., Shachnai, H., Tamir, T.: Maximizing submodular set functions subject to multiple linear constraints. In: 20th ACM-SIAM Symposium on Discrete Algorithms, pp. 545–554. Society for Industrial and Applied Mathematics, Philadelphia, PA (2009)
22. Lee, J., Mirrokni, V.S., Nagarajan, V., Sviridenko, M.: Maximizing non-monotone submodular functions under matroid or knapsack constraints. SIAM J. Discrete Mathematics 23(4), 2053-2078 (2010)
23. Lee, J., Sviridenko, M., Vondrák, J.: Submodular maximization over multiple matroids via generalized exchange properties. In: 12th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems, pp. 244–257. Springer-Verlag, Heidelberg (2009)
24. Lindley, D.V.: Dynamic programming and decision theory. Applied Statistics 10, 39–51 (1961)
25. Vondrák, J.: Symmetry and approximability of submodular maximization problems. In: 50th Annual IEEE Symposium on Foundations of Computer Science, pp. 651–670. IEEE Computer Society, Washington DC (2009)

A Example - the Classical Secretary Problem

The Classical Secretary Problem (CS) is a secretary problem with the set $\mathcal{I}$ of independent sets consisting of all singletons. We demonstrate the usefulness of the continuous model by analyzing an algorithm for this problem.

CS Algorithm(f)**:**

1. Observe the secretaries arriving till time $t = e^{-1}$, and let L be the set of secretaries arriving until this time.
2. Let $\hat{s}$ be the maximum secretary in L with respect to $\succ_\varnothing$.[a]
3. After time t, accept the first secretary s such that $f(s) \succ_\varnothing f(\hat{s})$.

[a] If no secretary arrives till time t, we assume $s \succ_\varnothing \hat{s}$ for every secretary s.

Theorem 7. *The above algorithm for CS is e^{-1}-competitive.*

Proof. Let s^* be the secretary of the optimal solution (breaking ties in favor of the earlier secretary according to $\succ_\varnothing$). Given that s^* arrives at some time $x \in (t, 1)$, s^* is accepted if one of the following conditions hold:

– No secretary arrives before time x.
– The best secretary arriving in the time range $[0, x)$ arrives before time t.

Since the secretaries are independent, with probability at least t/x, at least one of these conditions holds. The probability that s^* arrives in an interval of size ℓ is ℓ. Hence, the probability it arrives in an infinitesimal interval of size dx is dx. Therefore, by the law of total probability, the probability that the above algorithm accept s^* is at least

$$\int_t^1 \frac{t}{x} dx = t[\ln x]_t^1 = -t \ln t = e^{-1}.$$