Secretary Problems: Weights and Discounts

Moshe Babaioff* Michael Dinitz† Anupam Gupta† Nicole Immorlica‡ Kunal Talwar§

Abstract

The classical secretary problem studies the problem of selecting online an element (a "secretary") with maximum value in a randomly ordered sequence. The difficulty lies in the fact that an element must be either selected or discarded upon its arrival, and this decision is irrevocable. Constant-competitive algorithms are known for the classical secretary problems (see, e.g., the survey of Freeman [7]) and several variants. We study the following two extensions of the secretary problem:

- In the *discounted secretary problem*, there is a time-dependent "discount" factor $d(t)$, and the benefit derived from selecting an element/secretary e at time t is $d(t) \cdot v(e)$. For this problem with arbitrary (not necessarily decreasing) functions $d(t)$, we show a constant-competitive algorithm when the expected optimum is known in advance. With no prior knowledge, we exhibit a lower bound of $\Omega(\frac{\log n}{\log \log n})$, and give a nearly-matching $O(\log n)$-competitive algorithm.
- In the *weighted secretary problem*, up to K secretaries can be selected; when a secretary is selected (s)he must be irrevocably assigned to one of K positions, with position k having weight $w(k)$, and assigning object/secretary e to position k has benefit $w(k) \cdot v(e)$. The goal is to select secretaries and assign them to positions to maximize $\sum_{e,k} w(k) \cdot v(e) \cdot x_{ek}$ where x_{ek} is an indicator variable that secretary e is assigned position k. We give constant-competitive algorithms for this problem.

Most of these results can also be extended to the *matroid secretary* case (Babaioff et al. [2]) for a large family of matroids with a constant-factor loss, and an $O(\log \text{rank})$ loss for general matroids. These results are based on a reduction from various matroids to partition matroids which present a unified approach to many of the upper bounds of Babaioff et al. These problems have connections to online mechanism design (see, e.g., Hajiaghayi et al. [9]). All our algorithms are monotone, and hence lead to truthful mechanisms for the corresponding online auction problems.

1 Introduction

The classical secretary problem [5, 7] captures the question of finding the element with the maximum value in an *online* fashion, when the elements are presented in a random order. It is well known that waiting until one sees $1/e$ fraction of the elements, and picking the first element attaining a value greater than the maximum value seen in the first $1/e$ fraction of the elements gives an e-competitive algorithm, and this is the best possible. The problem is of interest due to its connections with online mechanism design: if we have a single good to sell and agents with varying valuations for that object arriving online (albeit in a random order[1]), then the secretary problem captures the difficulty in picking the person with the largest valuation for that good [9, 10]. In this case, the elements of the secretary problem are agents and the element value is the agent's value for the good; the goal of the mechanism designer is to *maximize social welfare*, or sell the good to the agent with the highest valuation. Another application is to modeling the economic decision facing an agent who wishes to select one of an online sequence of goods—e.g., an agent buying a house or a company hiring an employee. In this case, the elements are the goods and the element value is the value of the agent for the said good.

Given the above interpretations, there are certain natural cases which the secretary problem does not address: e.g., it does not capture the opportunity cost incurred due to delay in selecting an element. For example, when seeking to purchase a house, we might think of choosing a slightly suboptimal house at the beginning of the experiment (and being able to occupy it for the entire period) as being more desirable than a long wait to pick the most desirable house. We model such a problem as the *discounted secretary problem*, where we are given "discount" values $d(t)$ for every time step t: the benefit derived from choosing an element with value $v(e)$ at time t is the product $d(t) \cdot v(e)$. In this example, the discount function $d(t)$ is a monotone decreasing function of t, but in general the discount function may be more complicated due to other considerations. For example, our financial situation may improve over time and waiting longer may get us a better mortgage rate, so our "discount" function $d(t)$ may increase up to some point in time, and then decrease.[2]

An orthogonal extension of the classical secretary problem is the *weighted secretary problem*. In this case, there are K heterogeneous goods $\{1, 2, \ldots, K\}$, with the k^{th} good having a publicly known "weight" $w(k)$ (with higher

*moshe@microsoft.com. Microsoft Research, Silicon Valley

†Computer Science Department, Carnegie Mellon University, Pittsburgh, PA 15213. Research partly supported by NSF awards CCF-0448095 and CCF-0729022, and an Alfred P. Sloan Fellowship.

‡immorlic@cwi.nl. Centrum voor Wiskunde en Informatica (CWI), Amsterdam.

§kunal@microsoft.com. Microsoft Research, Silicon Valley

[1]The random order assumption allow us to overcome the impossibility of achieving good competitive ratio in the case of adversarial input without assuming that values come from some distribution.

[2]In this paper, we use the term "discount" though the function $d(t)$ is not necessarily monotone decreasing as a function of t.

numbers indicating greater desirability), such that if agent e has an "intrinsic value" $v(e)$ for good k, then assigning k to agent e accrues an actual value of $v(e) \cdot w(k)$. Such "product valuations" are commonly assumed in industries like online banner advertising, where goods are banner advertising space, the weight of the good is its visibility or the number of people that are likely to see it, and the intrinsic value of the advertising company is the value it derives when one person sees its ad. Similarly, product valuations might be observed in hiring scenarios: e.g., a company may wish to hire sales managers for several regional markets of varying sizes. The weights of the goods (job positions, in this case) are the market sizes, and the value of a manager is his or her inherent ability to convert peoples' interest into actual sales. Again, if the elements arrive in a random order, and assigning good k to agent e accrues a benefit of $v(e) \cdot w(k)$, how should we choose K agents and assign goods to them to maximize the total expected benefit?

1.1 Our Results

Surprisingly, the discounted secretary problem is interesting even if we know the values of all the items in advance: given discount function $d(t)$, it is not *a-priori* obvious which item the online algorithm should choose (we prove a constant lower bound, see Theorem 4.4). Our first theorem is the following.

THEOREM 1.1. (DISCOUNTED SEC'Y: KNOWN-OPT) *The discounted secretary problem has an $O(1)$-competitive algorithm for the case when the values of all elements are known in advance. In fact, the algorithm only needs to know* $\mathbf{E}[\mathsf{OPT}]$.[3]

The assumption that the values of element are known holds, say, when the value is a function of the ordinal preference (e.g., the value of the top candidate among n candidates is n, the second-best candidate has value of $n-1$, and so on), which has been used in [14]. Alternatively, $\mathbf{E}[\mathsf{OPT}]$ may be estimated by market research into prior experiments (which is a more realistic assumption in the mechanism design framework). Our next result shows that the knowledge assumption is essential for constant-competitive algorithms:

THEOREM 1.2. (DISCOUNTED SEC'Y: UNKNOWN-OPT) *Any algorithm for discounted secretary has a (worst-case) competitive ratio of $\Omega(\frac{\log n}{\log\log n})$. Moreover, there is a nearly matching $O(\log n)$-competitive algorithm.*

For the weighted secretary problem, we show the following.

THEOREM 1.3. (WEIGHTED SEC'Y) *There is a 4-competitive algorithm for the weighted secretary problem.*

[3]$\mathbf{E}[\mathsf{OPT}]$ is the expected benefit the optimal algorithm gets.

As the classical secretary problem is a special case of the weighted secretary problem when there is only one non-zero weight, there is clearly a lower-bound of e for the weighted secretary problem. In the setting with *both discounts and weights*, we show that a combination of the above algorithms yields a nearly-optimal result (since the $\Omega(\frac{\log n}{\log\log n})$ lower bound still holds).

THEOREM 1.4. (DISCOUNTED WEIGHTED SEC'Y) *There is an $O(\log n)$-competitive algorithm for the secretary problem with both weights on goods and discounts on times.*

Finally, we consider the discounted and weighted versions of the *matroid secretary problem* [2], where the goal is to choose a set of items in order to maximize the total expected value, subject to the constraint that the chosen set is independent in a given matroid (see Section 5 for definitions). We show that the algorithms of Theorems 1.1, 1.2 and 1.3 can be extended to a large family of matroids (including uniform matroids, partition matroids, graphical matroids) with only a constant loss in the competitive ratio, and to all matroids with an $O(\log \mathrm{rank})$ loss in the competitive ratio. These results are based on reductions from various classes of matroids to partition matroids, which also give a unified approach to many of the upper bounds of [2], whilst improving some of them. For example, our techniques imply the following:

THEOREM 1.5. *There is a $3e \approx 8.15$-competitive algorithm for the matroid secretary problem for graphical matroids. (The previous best known was a 16-competitive algorithm [2], and recently a $2e \approx 5.44$-competitive algorithm has been developed [11])*

While we state our results as algorithms, the setting which motivates us is actually an economic one in which the elements are strategic agents and their values are private information. In this case, it is important to consider the incentives facing agents in our proposed algorithms. Assuming single parameter agents (i.e., agent e has value $v(e)$ if he is picked by the algorithm, and 0 otherwise), it is well known that a mechanism is truthful (in dominant strategies) if it is bid monotonic (a winner would keep winning if she increases her bid). An alternative interpretation is that the mechanism presents the agent with a price that is independent of the agent's bid and the agent decides if she would like to win given the price. This is indeed the case for all our algorithms, thus one can interpret our algorithms as truthful mechanisms in which winners pay the threshold value they needed to bid in order to win. [4] The competitive ratio that an algorithm achieves corresponds to the fraction of the social welfare that the truthful mechanism guarantees.

Related Work. The study of truthful on-line mechanism design in the competitive analysis framework was

[4]Note that we do not allow agents to manipulate their arrival time - their order is random and cannot be influenced by the agents. Moreover, each agent is considered only once and cannot return later in time.

initiate by Lavi and Nisan [12] and many other papers followed this line of research, e.g. [13]. The classical secretary problem was first studied by Lindley [14] and by Dynkin in 1963 [5]. Since then, many variants have been studied (see [6, 7] for a survey), including some in the computer science literature highlighting the connections to online mechanism design [9, 10] and combinatorial preference structures [2, 1]. Recently, Dimitrov and Plaxton [4] gave a 16-competitive algorithm for the secretary problem on transversal matroids, improving on a result of [2], and then Korula and Pál [11] improved the analysis to show that it was in fact 8-competitive. The discounted problem was studied previously in multiple contexts for specific "well-behaved" functions like $d(t) = \beta^t$ [16] or $d(t) = \sum_{i=t}^{n} \beta^t$ [15] for some fixed $\beta < 1$, whereas here we study it for general functions $d(t)$. For the weighted case, Derman, Lieberman, and Ross [3] studied a version where there are the same number of goods as agents (so $K = n$) and the values of the agents are independently and identically distributed, as opposed to our setting where the values are arbitrary but arrive in random order. Similarly, a recent paper by Gershkov and Moldovanu [8] studied the variant where the values of the elements are independently and identically distributed and element arrivals are given by some renewal stochastic process, instead of the classical secretary assumption of discrete time with a uniformly random ordering. They further incorporated the discounted model into the weighted model (so the value of agent e for good k at time t is $v(e) \cdot w(k) \cdot d(t)$, and show how to maximize revenue as well as welfare in their setting).

2 Model, Preliminaries, and Notation

In the classical secretary problem, there is a universe U of secretaries/elements with $|U| = n$. Each secretary/element $e \in U$ has an intrinsic value $v(e) \in \mathbb{R}_{\geq 0}$. An algorithm A for the secretary problem observes the elements of U in a random order and chooses one element e_A in an *online* fashion. In other words, it must decide at the moment an element arrives whether or not to select it, and all decisions are irrevocable. The goal is to maximize the expected value $\mathbf{E}[v(e_A)]$, the expectation being taken over the random order, as well as over the randomness in the algorithm, if any. In this paper, we extend the classical setting as follows:

Weighted Secretary Problems. Here, we want to choose K secretaries and match them to one of K positions. Each position k has a non-negative weight $w(k)$ with $w(1) \geq w(2) \geq \cdots \geq w(K)$, each secretary e has an intrinsic value $v(e)$. Our algorithm A must build online the assignment map $s_A : [K] \to U \cup \{\perp\}$, where we interpret $s_A(k) = e$ as meaning secretary e is assigned to position k, and $s_A(k) = \perp$ as meaning that no secretary has been assigned to the kth position. (Initially $s(k) = \perp$ for all $k \in [K]$.) For ease of notation we extend the agent valuation function by letting $v(\perp) = 0$. When the secretary e arrives the algorithm finds out $v(e)$; if it decides to choose secretary e, it must immediately and irrevocably assign e to some unassigned position k (and hence set $s_A(k) = e$). The goal of the algorithm is to output a final map s_A that maximizes $\mathbf{E}\left[\sum_{k=1}^{K} v(s_A(k))w(k)\right]$, where the expectation is over the random ordering and the random choices of the algorithm.

Discounted Secretary Problems. Here, the algorithm is given as input a discount function $d : [n] \to \mathbb{R}_{\geq 0}$ which maps "time" to "discounts". Again, the ground set is presented in random order: we formalize this as picking a bijective ordering function $\pi : [n] \to U$ uniformly at random from all bijective functions from time instants $[n]$ to elements U, implying that element $\pi(t) \in U$ appears at time instant t. In this problem, we want to choose an element e_A in an online fashion to maximize the expected discounted value $\mathbf{E}_\pi[d(\pi^{-1}(e_A)) \cdot v(e_A)]$. In the *known*-$\mathsf{OPT}$ *model*, the algorithm knows the expected value of the optimal offline solution ahead of time (for example, it is sufficient to assume the algorithm knows the valuations, see Lemma A.1), it just does not know the random permutation π—whereas in the *unknown*-OPT *model*, the algorithm does not have any such prior knowledge about the input. For this problem, since the optimum offline solution is itself a random variable, a function of the random order π of arrivals, we consider the competitive ratio of an algorithm A to be the smallest value α such that

$$\frac{\mathbf{E}_\pi[\max_e d(\pi^{-1}(e)) \cdot v(e)]}{\mathbf{E}_\pi[d(\pi^{-1}(e_A)) \cdot v(e_A)]} \leq \alpha.$$

It is well-known that the following algorithm is e-competitive for the classical secretary problem: observe a $(1/e)$ fraction of the elements without selecting any. In the remaining $(1 - 1/e)$ fraction of elements, select the first element whose value is greater than all elements preceding it. In the remainder of the paper, we will refer to this algorithm as the *classical secretary algorithm*, and we will use it (and the sample-then-select intuition behind it) to design algorithms for the weighted and discounted cases.

Finally, all the secretary problems mentioned above can be extended to the *matroid secretary case* where, instead of picking a set S of one or K elements, we are given a matroid $\mathcal{M} = (U, \mathcal{I})$ and want to pick a set $S \in \mathcal{I}$ that maximizes the objective function. Details of this extension, and background on matroids, appear in Section 5.

3 The Weighted Secretary Problem

Recall the weighted secretary problem was to choose, in an online fashion, up to K secretaries and assign each one irrevocably to one of K positions (with weights $w(1) \geq w(2) \geq \cdots \geq w(K)$) so that no position has more than one secretary. The goal was to maximize the weighted value $\sum_{k=1}^{K} v(s(k)) \cdot w(k)$, where $s(k)$ is the secretary that is assigned to position k and $v(s(k)) = 0$ if position k is not filled by any secretary (i.e. $s(k) = \perp$). In this section, we show a constant-competitive algorithm for this problem. We assume, without loss of generality, that the values of the secretaries are all distinct.

We will use the following algorithm, called the *interval reservation algorithm*:

a. Observe the first $l = \lfloor \frac{n}{2} \rfloor$ secretaries in the random sequence without assigning any of them. Call these secretaries the *sample set* S.

b. Compute the optimal solution on the sample set, which will just fill position i with the ith most valuable secretary in S. For each position k let a_k be the value of the secretary that fills it, and for $k > 1$ let $I(k)$ be the interval (a_k, a_{k-1}). For position 1, let $I(1) = (a_1, \infty)$. Note that all of these intervals are disjoint.

c. Now consider the remaining $(n - l)$ secretaries. When secretary e arrives, let k_e be such that $v(e) \in I(k_e)$; if there is at least one free position $k \geq k_e$, then assign secretary e to the position k with the lowest such index.

To analyze this algorithm, we consider a slightly different algorithm, where in step **c** above, when considering secretary e, we assign it only to position k_e if it is free, else we discard it. (Let us call this *Algorithm B*. Note that while the interval reservation algorithm is monotone[5], Algorithm B may not be monotone.)

LEMMA 3.1. *The expected weighted value achieved by the interval reservation algorithm is at least that achieved by Algorithm B.*

Proof. Consider a position k. For any fixed permutation, it is immediate that position k is filled in the interval reservation algorithm by an agent with value at least as large as the agent filling position k in Algorithm B. The claim follows.

LEMMA 3.2. *If a secretary e is assigned to some position by Algorithm B, then the weight of this position is at least as large as the weight of the position e is assigned to in the optimal solution (if any).*

Proof. Suppose secretary e is assigned to position k by Algorithm B. Then $v(e)$ was not in $I(s)$ for $s < k$, so there were $k - 1$ secretaries in the sample set with value greater than $v(e)$. Since the optimal solution fills position i with the ith most valuable secretary, and there are at least $k - 1$ secretaries more valuable than e, the optimal solution will not assign e to a position with weight greater than $w(k)$.

Hence, as long as any secretary chosen in the optimal solution is also chosen by Algorithm B with reasonable probability, we are fine. The next lemma shows exactly this:

[5]We assume that agent i's value from being assigned position k is $v(i) \cdot w(k)$, where $v(i)$ is her private valuation. Incentive compatibility then requires that the algorithm be (weakly) monotone, i.e. if agent i increases her bid, the expected weight of the position she is assigned does not decrease.

LEMMA 3.3. *If a secretary is assigned to some position by the optimal solution, then the probability that it is assigned to some position by Algorithm B is at least $1/4$.*

Proof. Consider a secretary e that is assigned to some position by the optimal solution. Given that e appears in the random order at position $j \geq n/2$, then with probability $\frac{l}{j-1}$ the next more valuable secretary that appeared before e actually appears in the sample. Conditioned on this event, with probability $\frac{l-1}{j-2}$ the next less valuable secretary that appeared before e also appeared in the sample. This is a sufficient condition for e to be assigned a position, since it implies that there's some position k whose interval $I(k)$ contains $v(e)$ and which will not have been filled by the time that e appears. Thus the probability that secretary e is assigned to some position is at least

$$\sum_{j=l+1}^{n} \frac{1}{n} \times \frac{l}{j-1} \times \frac{l-1}{j-2} = \frac{l(l-1)}{n} \sum_{j=l+1}^{n} \frac{1}{(j-1)(j-2)} =$$

$$\frac{l(l-1)}{n} \sum_{j=l+1}^{n} \left(\frac{1}{j-2} - \frac{1}{j-1}\right) = \frac{l(n-l)}{n(n-1)}.$$

This expression is at least $\frac{1}{4}$ when $n \geq 4$. It is easy to see that the Algorithm B also satisfies this lemma for $n < 4$, thus proving the lemma.

Lemmas 3.2 and 3.3, together with linearity of expectations, imply the following theorem.

THEOREM 3.1. (WTD SECRETARY: UPPER BOUND) *Algorithm B is 4-competitive. Hence, by Lemma 3.1, the interval reservation algorithm is 4-competitive.*

4 Time-dependent Weights, or the Discounted Secretary Problem

In the discounted secretary problem, we are given a function $d(\cdot)$ that maps time instants to "discounts", such that the actual benefit obtained by picking an element e of intrinsic value $v(e)$ at time t is actually the product $v(e) \cdot d(t)$. This is a natural model when picking an item is more valuable at some times rather than others: while the problem has been studied in the simple case of $d(t) = \beta^t$ for some $\beta < 1$, here we consider general *time-varying* functions $d(t)$.

We first show that the problem is fairly hard in general. We show an $\Omega(\log n / \log \log n)$ lower bound on the competitive ratio of any algorithm, and show a nearly matching $O(\log n)$ upper bound. Surprisingly, the problem becomes much easier with a small amount of information about the input. We show that knowing an estimate of $\mathbf{E}[\mathsf{OPT}]$ enables one to design a constant competitive algorithm. We remark that all our algorithms are monotone, and thus can be converted to truthful mechanisms.

A Warmup Example. Consider the simple case when $d(t) = \beta^t$ for some constant β bounded away from 1. A simple constant-competitive algorithm is one that always picks the first element. This algorithm has an expected value of $\mathbf{E}[\mathsf{ALG}] = \sum_{i=1}^{n} v(i) \cdot \frac{1}{n} \cdot \beta^1$. On the other hand, the expected value that OPT gets from time step j is at most $\sum_{i=1}^{n} v(i) \cdot \frac{1}{n} \cdot \beta^j$, thus the expected value that OPT gets is at most $\sum_{i=1}^{n} v(i) \cdot \frac{1}{n} \cdot \sum_j \beta^j \leq \frac{1}{1-\beta}\mathbf{E}[\mathsf{ALG}]$. As β is a fixed constant this is constant-competitive.

4.1 Discounted Secretary: General Case

It turns out that the discounted secretary problem is significantly harder in the *unknown-*OPT *model*, and we show a lower bound of $\Omega(\frac{\log n}{\log\log n})$ on the competitive ratio in this case, even when the discounts are decreasing. However, we also give an algorithm with an almost-matching competitive ratio of $O(\log n)$, for arbitrary discounts.

4.1.1 A Nearly-Logarithmic Lower Bound We construct a discount function d and a family of instances $\mathcal{I}_1, \ldots, \mathcal{I}_{2c}$ such that no randomized algorithm can be $\Theta(\frac{\log n}{\log\log n})$-competitive on all the $\mathcal{I}_t$'s. More formally, we use the following definitions:

- **Instance Size:** Let $L = c$ be an integer and let $n = L^{4c}$. Thus $L = c = \Theta(\frac{\log n}{\log\log n})$.
- **Discount Function:** For $t \leq 2c$, let $n_t = L^{2t}$. For the discount function, we use a step function $d(j) = L^{-t}$ for $n_{t-1} \leq j < n_t$.
- **Instances:** Let K be a large enough integer (say n^2). The instance $\mathcal{I}_1$ consists of $\frac{n}{n_1}$ K's and the remaining zeroes. For $t < 2c$, $\mathcal{I}_{t+1}$ is obtained inductively from $\mathcal{I}_t$ by changing n_t/n_{t+1} of the K^t's to K^{t+1}. Thus $\mathcal{I}_t$ has $\frac{n}{n_t}$ K^t's.

LEMMA 4.1. (ESTIMATE ON $\mathbf{E}[\mathsf{OPT}(\mathcal{I}_t)]$) *For* $t \leq 2c$, $\mathbf{E}[OPT(\mathcal{I}_t)]$ *is at least* $(1 - \frac{1}{e})K^tL^{-t}$

Proof. With probability at least $(1 - \frac{1}{e})$ over the random permutation, one of the n/n_t K^t values falls in the first n_t slots, leading to a value of at least K^tL^{-t}.

Let $\mathcal{A}$ be a $c/10$-competitive algorithm. We argue that $\mathcal{A}$ must have a large probability of picking an item in each of the intervals (n_t, n_{t+1}).

LEMMA 4.2. *Let $\mathcal{A}$ be a $c/10$-competitive algorithm. Then on instance $\mathcal{I}_t$, the probability that $\mathcal{A}$ picks one of the first n_t items is at least t/c.*

Proof. We prove the claim by induction on t. For the base case, note that the most $\mathcal{A}$ can get from time steps $n_1 + 1$ onwards is $K/L^2 = \frac{1}{c}(\frac{K}{L})$. Thus to be $\frac{c}{10}$ competitive, it must in expectation get value at least $\frac{2}{c}(\frac{K}{L})$ from the first n_1 time steps. Since $\mathcal{A}$ never gets more than K/L on $\mathcal{I}_1$, the base case holds.

Assume that the claim holds for $\mathcal{I}_t$. Consider a run of the algorithm on instance $\mathcal{I}_{t+1}$. Note that $\mathcal{I}_{t+1}$ differs from $\mathcal{I}_t$ only in $\frac{n}{n_{t+1}}$ of the items. The probability that any one of these items occurs in the first n_t time steps is at most $\frac{n_t}{n_{t+1}}$. Thus except with probability $\frac{n_t}{n_{t+1}} = \frac{1}{L^2}$, the behavior of $\mathcal{A}$ on $\mathcal{I}_t$ and $\mathcal{I}_{t+1}$ is indistinguishable in the first n_t steps. Thus by the induction hypothesis, the algorithm accepts an item by time n_t with probability at least $(1 - \frac{1}{L^2})(t/c)$.

The expected revenue of $\mathcal{A}$ in the first n_t time steps, on instance $\mathcal{I}_{t+1}$ is bounded by $K^{t+1}(L^{-t}\frac{n_t}{n_{t+1}} + L^{-(t-1)}\frac{n_{t-1}}{n_{t+1}} + \ldots + L^{-1}\frac{n_1}{n_{t+1}}) + K^tL^{-1}$, where the first term bounds the expected contribution from the K^{t+1}'s, and the second term bounds the most one can get from the smaller items. Since this is at most $4OPT/c$, $\mathcal{A}$ must get at least $6OPT/c$ from time steps $n_t + 1$ onwards. As in the base case, time steps $n_{t+1} + 1$ and higher cannot contribute more than $2OPT/c$, so that the algorithm must get contribution $4OPT/c$ from time steps $[n_t + 1, n_{t+1}]$. Thus the probability that the algorithm picks an item in time steps $[n_t + 1, n_{t+1}]$ is at least $\frac{2}{c}$. Since $\mathcal{A}$ picks exactly one item, this event is disjoint from $\mathcal{A}$ picking an item in time steps $[1, n_t]$. Thus the probability that $\mathcal{A}$, on instance $\mathcal{I}_{t+1}$ picks an item in times steps $[1, n_{t+1}]$ is at least $(1 - \frac{1}{L^2})(t/c) + (2/c) \geq (t+1)/c$. The claim follows.

Now we can prove the main lower bound theorem.

THEOREM 4.1. (UNKNOWN-OPT: LOWER BOUND) *Every algorithm $\mathcal{A}$ for the discounted secretary problem has* $\mathbf{E}[\mathsf{OPT}]/\mathbf{E}[\mathcal{A}] \geq \Omega(\log n/\log\log n)$ *in the worst case.*

Proof. If $\mathcal{A}$ is $c/10$-competitive, then Lemma 4.2 implies that on instance $\mathcal{I}_{2c}$, the probability that $\mathcal{A}$ picks an item in the first n_{2c} time steps is larger than 1, which is a contradiction. Thus $\mathcal{A}$ cannot be $c/10$-competitive.

4.1.2 A Logarithmic Upper Bound Let d_{max} be the maximum discount and let v_{max} be the maximum value. For $c \geq 1$ let $I_c = (2^{-c}d_{max}, 2^{-(c-1)}d_{max}]$ be the interval defining the c-th discount class, and let $P_c = \{i \in [n] : d(i) \in I_c\}$ be the set of times that have discount value in class c. Our algorithm $\mathcal{A}$ chooses a $c \in [3\log n + 2]$ uniformly at random, and then runs the classical secretary in the time steps $i \in P_c$. Note that $\mathcal{A}$ uses no knowledge of $\mathbf{E}[\mathsf{OPT}]$.

THEOREM 4.2. (UNKNOWN-OPT: UPPER BOUND) *Algorithm $\mathcal{A}$ is $O(\log)$-competitive, i.e.* $\mathbf{E}[\mathsf{OPT}]/\mathbf{E}[\mathcal{A}] \leq O(\log n)$.

Proof. We begin by noting that $\mathbf{E}[\mathsf{OPT}] = \sum_c \sum_{i \in P_c} d(i) \sum_j v(j) Pr[(\pi(i) = j) \wedge (d(i)v(j) \geq d(k)v(\pi(k))\ \forall k \in [n])]$. Let OPT_c denote the term corresponding to P_c so that $\mathbf{E}[\mathsf{OPT}] = \sum_c \mathsf{OPT}_c$. Clearly $\mathsf{OPT}_1 \geq v_{max}d_{max}/n$, since v_{max} occurs in the location corresponding to d_{max} with probability at least $\frac{1}{n}$. Moreover, OPT_c equals

$$\begin{aligned} & \textstyle\sum_{i\in P_c} d(i)\sum_j v(j)Pr[(\pi(i)=j)\wedge \\ & \quad (d(i)v(j)\geq d(k)v(\pi(k))\ \forall k\in[n])] \\ \leq\ & \textstyle\sum_{i\in P_c} 2^{-(c-1)}d_{max}\sum_j v(j) \\ \leq\ & 2^{-(c-1)}d_{max}|P_c|nv_{max} \end{aligned}$$

since each $v(j)$ is bounded by v_{max}. And since $|P_c|$ is at most n, we conclude that $\mathsf{OPT}_c \leq 2^{-c}2n^2d_{max}v_{max}$. It follows that $\sum_{c\geq 3\log n+2}\mathsf{OPT}_c \leq \mathbf{E}[\mathsf{OPT}]/2$, so that $\sum_{c=1}^{3\log n+1}\mathsf{OPT}_c \geq \mathbf{E}[\mathsf{OPT}]/2$. Thus OPT gets most of its value for the top $O(\log n)$ discount scales.

Clearly the expected value that the offline optimum gets from time steps in P_c equals OPT_c. Thus the expected value of the offline optimum for the problem restricted to times in P_c is at least as large, and the classical secretary algorithm gets an expected value $\mathsf{OPT}_c/2e$ (we lose an additional factor of two since we treat the discount values in P_c equally). The claim follows.

When there are both weights and discounts, we can combine Theorem 4.2 with Theorem 3.1 to get the following result. We defer the proof to the full version.

THEOREM 4.3. *There is an algorithm for the weighted discounted secretary problem (which uses no knowledge of* $\mathbf{E}[\mathsf{OPT}]$*) that is* $O(\log n)$*-competitive*

Proof Sketch: In the unweighted case the proof of Theorem 4.2 implies that we can ignore all but the first $\Theta(\log n)$ discount classes. Using the same reasoning, in the weighted case we can ignore all but the first $\Theta(\log(nK))$ discount classes, where the extra K is due to the ability to assign K goods. Now we can just choose a $c\in[\Theta(\log(nK))]$ uniformly at random and run the weighted secretary algorithm of Theorem 3.1 in the time steps with discount in class c. Following the analysis of Theorem 4.2, this is an $O(\log(nK))$-competitive algorithms, and thus is also $O(\log n)$-competitive (as $K\leq n$). ■

4.2 Discounted Secretary with Known OPT

In this section we consider the case when the algorithm knows a good estimate for $\mathbf{E}[\mathsf{OPT}]$ ahead of time. Interestingly, we can show that even if we know *all the values* and not just $\mathbf{E}[\mathsf{OPT}]$ in advance, no online algorithm can be perfectly competitive. This is in contrast to the case without discounts (i.e., $d(t)=1\forall t$), in which the naïve algorithm that knows all the values can pick an element of maximal value.

THEOREM 4.4. (KNOWN-OPT: LOWER BOUND) *For any* $\epsilon > 0$*, every algorithm* $\mathcal{A}$ *for the discounted secretary problem has* $\mathbf{E}[\mathsf{OPT}]/\mathbf{E}[\mathcal{A}]\geq\sqrt{2}-\epsilon$*, even when* $\mathcal{A}$ *knows the set of values (and hence* $\mathbf{E}[\mathsf{OPT}]$*) in advance.*

Proof. Let $c=\sqrt{2}-1$. Let $t=\lfloor n(1-c)\rfloor$, and note that $nc\leq n-t\leq nc+1$. Assume that $v(1)=v(2)=1$ and that $v(i)=0$ for any $i\geq 3$. Let $d(i)=1$ for $i<t$, $d(t)=c\cdot n+1$ and $d(i)=0$ for any $i>t$. The expected value that OPT gets is $\mathbf{E}[\mathsf{OPT}]=1*(1-(\frac{n-t}{n})^2)+c\cdot n\cdot\frac{2}{n}\geq 1-(\frac{nc+1}{n})^2+2c=1+2c-c^2-\frac{2cn+1}{n^2}=4(\sqrt{2}-1)-\frac{2(\sqrt{2}-1)n+1}{n^2}$.

We next show that if $\mathcal{A}$ observes one of the two valuable elements (with value 1) before time t it immediately picks it (this is the optimal online algorithm for this input). This is true as the expected value from picking the element is 1. On the other hand, if there are still k element to come, the expected value from skipping the current element and picking the next valuable element is $1\cdot\frac{k-(n-t)}{k}+cn\cdot\frac{1}{k}\leq\frac{k-cn}{k}+cn\cdot\frac{1}{k}=1$. Thus the algorithm is always (weakly) better off picking the first valuable element it sees.

Now, the expected value that $\mathcal{A}$ gets is $\mathbf{E}[\mathcal{A}]=1*(1-(\frac{n-t}{n})^2)+cn\cdot2\cdot\frac{n-t}{n}\cdot\frac{1}{n}\leq 1-(\frac{cn}{n})^2+2c\frac{cn+1}{n}=1+c^2+\frac{2c}{n}=4-2\sqrt{2}+\frac{2(\sqrt{2}-1)}{n}$. We conclude that

$$\mathbf{E}[\mathsf{OPT}]/\mathbf{E}[\mathcal{A}]\geq\frac{4(\sqrt{2}-1)-\frac{2(\sqrt{2}-1)n+1}{n^2}}{4-2\sqrt{2}+\frac{2(\sqrt{2}-1)}{n}}$$

As this expression clearly tends to $\frac{4(\sqrt{2}-1)}{4-2\sqrt{2}}=\sqrt{2}$ as n goes to infinity, the claim follows.

4.2.1 The Upper Bound for Known-OPT

The main result of this section shows that one can get good competitive algorithms if we have a good estimate Z of $\mathbf{E}[\mathsf{OPT}]=\mathbf{E}_\pi[\max_{t\in[n]}d(t)\cdot v(\pi(t))]$.

Algorithm $\mathcal{A}$: Suppose $Z\leq\mathbf{E}[\mathsf{OPT}]$. Pick the first element e seen (say, at time j) that satisfies $v(i)d(j)\geq Z/2$.

THEOREM 4.5. (KNOWN-OPT: UPPER BOUND) *If* $Z\leq\mathbf{E}[\mathsf{OPT}]$*, the algorithm* $\mathcal{A}$ *for the discounted secretary problem satisfies* $\mathbf{E}[\mathcal{A}]\geq Z/4$.

Proof. The proof considers two cases. In the first (easier) case, suppose the algorithm $\mathcal{A}$ picks *some* element with probability at least $1/2$. Then with probability at least $1/2$, it gets benefit at least $Z/2$, and hence its expected performance is at least $Z/4$.

In the second case, suppose the algorithm $\mathcal{A}$ does not pick an element with probability at least $1/2$: i.e., we are in the case where at least half the permutations cause all products $\{v(e)\,d(\pi(e))\}_{e\in U}$ to be small (and hence are "rejecting"). In this case, we can show that OPT gets most of its benefit from the other, "accepting" permutations. Moreover, on these accepting permutations, we would pick an element differently from OPT only when there was some 'blocking' element with $v(e)\,d(\pi(e))\geq Z/2$: but for the same reason that there are few accepting permutations, there are few accepting permutations with such a blocking element. Hence we pick the same elements OPT picks on the "accepting" permutations with high probability, and have a good performance.

Let us now make this general idea for the second case precise. Let us write

$$\mathbf{E}[\mathsf{OPT}] = \sum_{\pi \in S_n} \frac{1}{n!} \max_{i=1}^{n}\{d(i)v(\pi(i))\}$$

Let $S_{acc} = \{\pi \in \mathfrak{S}_n : \max_{i=1}^n\{d(i)v(\pi(i)) \geq Z/2\}$ be the "accepting permutations" on which the algorithm picks some element. The contribution to $\mathbf{E}[\mathsf{OPT}]$ from these accepting permutations is

$$\begin{aligned} L &\overset{def}{=} \textstyle\sum_{\pi \in S_{acc}} \frac{1}{n!} \max_{i=1}^n\{d(i)v(\pi(i))\} \\ &= \mathbf{E}[\mathsf{OPT}] - \textstyle\sum_{\pi \notin S_{acc}} \frac{1}{n!} \max_{i=1}^n\{d(i)v(\pi(i))\} \\ &\geq \mathbf{E}[\mathsf{OPT}] - \textstyle\sum_{\pi \notin S_{acc}} \frac{1}{n!}(Z/2) \\ &\geq \mathbf{E}[\mathsf{OPT}] - \tfrac{Z}{2} \geq \tfrac{Z}{2}, \end{aligned} \tag{4.1}$$

where we used the fact that $Z \leq \mathbf{E}[\mathsf{OPT}]$. To complete the proof, it suffices to show that $\mathcal{A}$ has expected value at least $L/2$, which is at least $Z/4$ by (4.1). We begin by rewriting L as

$$\begin{aligned} L = \textstyle\sum_{i=1}^n \sum_{j: d(i)v(j) \geq \frac{Z}{2}} \frac{1}{n} \cdot d(i)v(j) \\ \times \mathbf{Pr}[d(i)v(j) \text{ highest } | \pi(i) = j] \end{aligned} \tag{4.2}$$

(Here we implicitly assume that there is exactly one product $d(i)v(j)$ which is highest: we break ties in some consistent fashion.) Combining (4.1) and (4.2) gives us

$$\textstyle\sum_{i=1}^n \sum_{j: d(i)v(j) \geq \frac{Z}{2}} \frac{1}{n} \cdot d(i)v(j) \geq \frac{Z}{2} \tag{4.3}$$

For each such i, j pair where $d(i)v(j) \geq Z/2$, let G_{ij} be the set of permutations on which $\mathcal{A}$ chooses element j at time i, getting benefit $d(i)v(j)$: let us call these "good". (Note that, in these permutations, $d(i)v(j)$ must be the *first* product which is at least $Z/2$.) The performance of the algorithm is

$$\mathbf{E}[\mathcal{A}] = \textstyle\sum_{i=1}^n \sum_{j: d(i)v(j) \geq \frac{Z}{2}} d(i)v(j) \cdot \frac{|G_{ij}|}{|\mathfrak{S}_n|} \tag{4.4}$$

The following claim shows there are many "good" permutations.

CLAIM 4.1. *For each i, j such that $d(i)v(j) \geq Z/2$, $|G_{ij}| \geq |\mathfrak{S}_n \setminus S_{acc}|/n \geq |\mathfrak{S}_n|/2n$.*

Proof. We show the first inequality by giving an n-to-1 map f_{ij} from $\mathfrak{S}_n \setminus S_{acc}$ to G_{ij}, the second inequality follows from the fact that at most half the permutations are in S_{acc}.

Define $f_{ij}(\pi)$ by changing π and mapping element j to location i, swapping it with whatever was there originally. In other words, if $i' = \pi^{-1}(j)$, let $f_{ij}(\pi)(i) = \pi(i') = j$, let $f_{ij}(\pi)(i') = \pi(i)$, and for $k \neq i, i'$ let $f_{ij}(\pi)(k) = \pi(k)$. Showing f_{ij} is a n-to-1 map is straight-forward; to show that $\pi' = f_{ij}(\pi)$ is in G_{ij}, we need to show that $d(i)v(\pi'(i)) = d(i)v(j) \geq Z/2$ (which follows from our choice of i, j), and that no other position i' has $d(i')v(\pi'(i')) \geq Z/2$ (which follows almost as easily from the fact that $\pi \notin S_{acc}$ and hence did not have any positions i' such that $d(i')v(\pi(i')) \geq Z/2$.

The rest of the proof of Theorem 4.5 is immediate: by (4.4) and Claim 4.1, it follows that

$$\mathbf{E}[\mathcal{A}] \geq \textstyle\sum_{i=1}^n \sum_{j: d(i)v(j) \geq \frac{Z}{2}} d(i)v(j) \cdot \frac{1}{2n} \geq Z/4,$$

where the last step is by equation (4.3).

To use Theorem 4.5 fruitfully and obtain benefit $\approx \mathbf{E}[\mathsf{OPT}]/4$, the bound Z should be close to $\mathbf{E}[\mathsf{OPT}]$: such an estimate could come from expert knowledge, prior runs of the problem, or some other source. A special case when such an estimate of $\mathbf{E}[\mathsf{OPT}]$ can be obtained is the situation when we know the values of all the elements. In this case, we can use a sampling-based estimator for $\mathbf{E}[\mathsf{OPT}]$ (as shown in Lemma A.1) to get $\mathbf{E}[\mathcal{A}] \geq \frac{(1-\delta)(1-\epsilon)}{4}\mathbf{E}[\mathsf{OPT}]$, where $\epsilon, \delta > 0$ are arbitrary constants that only affect the amount of sampling necessary.

5 Extensions to Matroid Secretary Problems

In this section, we show how the secretary problem on many classes of matroids can be "reduced" to partition matroids. Moreover, we extend our results for the discounted and weighted secretary problems to partition matroids, and hence to such classes of matroids as well. This reduction to partition matroids also gives a simple unified view of several results of Babaioff et al. [2], and also improves some of the results from their paper.

Recall that a matroid $\mathcal{M} = (U, \mathcal{I})$ consists of a ground set U, and a collection $\mathcal{I}$ of subsets of U that is closed under taking subsets satisfying the well-known exchange conditions[6]. A *partition matroid* $(U, \mathcal{I})$ is one where there is some partition P of U, and $S \in \mathcal{I}$ if and only if I has at most one element from each set in P. The following definition formalizes the notion of a "reduction" from arbitrary matroids to partition matroids.

DEFINITION 5.1. (AN α-PARTITION PROPERTY) *A matroid $\mathcal{M} = (U, \mathcal{I})$* satisfies an α-partition property *if one can (randomly) define a* partition matroid $\mathcal{M}' = (U', \mathcal{I}')$ *on some subset U' of the universe U such that for any values of the elements U, we have*

- *E(value of max-weight base in $\mathcal{M}'$) $\geq 1/\alpha \times$ value of max-weight base in $\mathcal{M}$.*
- *Every independent set in $\mathcal{M}'$ is an independent set in M.*

[6] i.e. for any $A, B \in \mathcal{I}$ with $|A| < |B|$, $\exists x \in B$ such that $A \cup \{x\} \in \mathcal{I}$.

In a *graphic matroid* the ground set is the set of edges of some graph $G = (V, E)$, and $S \subseteq E$ is independent if and only if it does not contain a cycle. In a *uniform matroid* of rank k the independent sets are all subsets of size at most k. Finally, in a *transversal matroid*, the ground set is the set of left vertices from some bipartite graph $G = (L, R, E)$, and a set $S \subseteq L$ is independent if and only if there is a perfect matching of S to nodes in R. All of these matroids satisfy some α-partition property:

THEOREM 5.1. *An α-partition property can be shown for the following classes of matroids:*

- *Any graphical matroid satisfies a 3-partition property.*
- *Any uniform matroid satisfies a $e/(e-1)$-partition property.*
- *Transversal matroids satisfy a d-partition property, where d is the maximum degree of vertices on the left, as well as a $4k/d$-partition property, where k is the rank of the matroid.*

The proofs for these reductions appear in Appendix B. Independent of our work, Korula and Pal recently proved that any graphical matroid satisfies a 2-partition property [11].

The reductions of Theorem 5.1 motivate the study of the matroid secretary problem, the weighted version, and the discounted version on partition matroids.

THEOREM 5.2. *On partition matroids, there is an e-competitive algorithm for the matroid secretary problem, a $(16+3e)$-competitive algorithm for the weighted matroid secretary problem, a $(\frac{(1-\delta)(1-\epsilon)}{4})$-competitive algorithm for the discounted matroid secretary problem if we know all of the values, and an $O(\log n)$-competitive algorithm for the discounted matroid secretary with unknown-*OPT.

Proof. The e-competitive algorithm for the matroid secretary problem is trivial: just run the classical secretary algorithm on each set in the partition. The same argument combined with Theorem 4.5 and Lemma A.1 gives the theorem for the discounted case with known-OPT, and combined with Theorem 4.2 gives the theorem for the discounted case with unknown-OPT. We defer these proofs, as well as the proof for the weighted problem, to the full version of the paper.

This theorem, combined with the definition of the α-partition property, immediately implies that if a matroid satisfies an α-partition property, then there is an $(e\alpha)$-competitive algorithm for the matroid secretary problem, a $((16+3e)\alpha)$-competitive algorithm for the weighted secretary problem, a $(\frac{(1-\delta)(1-\epsilon)}{4}\alpha)$-competitive algorithm for the discounted matroid secretary problem if we know all of the values, and an $O(\alpha \log n)$-competitive algorithm for the discounted matroids secretary with unknown-OPT. Thus Theorem 5.1 implies that there are constant-competitive algorithms for graphic, uniform, and low- and high-degree transversal matroids in all variants but the discounted unknown-OPT setting, in which case there are $O(\log n)$-competitive algorithms.

If we do not know that the matroid we are working with satisfies an α-partition property, we extend Theorem 4.2 to give the following bound for the discounted problem in the unknown-OPT model. (The proof is deferred to the full version of the paper.)

THEOREM 5.3. *There is an algorithm $\mathcal{A}$ for the discounted secretary problem on matroids with unknown-*OPT *such that* $\mathbf{E}[\mathsf{OPT}]/\mathbf{E}[\mathcal{A}] \leq O(\log k \cdot \log n)$ *where k is the rank of the matroid $\mathcal{M}$ which is the input to the algorithm.*

In Section 4.1 we saw that if one knows all the values, or knows (a good estimate of) the expectation of OPT this is enough to break the lower bound and achieve a constant competitive algorithm. For discounted secretary problem on matroids knowing the maximal value is also very helpful. In such case we can improve the result of Theorem 5.3.

THEOREM 5.4. *There is an algorithm $\mathcal{A}$ for the discounted secretary problem on matroids in case that the maximal value is known such that* $\mathbf{E}[\mathsf{OPT}]/\mathbf{E}[\mathcal{A}] \leq O(\log n)$.

Proof. For $c \in Z$, let scale P_c be the set of all pairs i, j such that $d(i)v(j) \in (2^{c-1}, 2^c]$. Thus $OPT \leq 2^c \sum_{(i,j)\in P_c} Pr[\pi(i) = j \wedge d(i)v(j) \in OPT]$. Since v_{max} falls in the maximum discount location with probability $\frac{1}{n}$, $OPT \geq v_{max}d_{max}/n$. Let $c_{max} = \lceil \log v_{max}d_{max} \rceil$ be the index of the highest non-empty scale. Since $|P_c| < n^2$, it follows that that contribution of level c to OPT, $2^c \sum_{(i,j)\in P_c} Pr[\pi(i) = j \wedge d(i)v(j) \in OPT]$ is bounded above by $n^2 2^c$. Thus for $c_{min} = c_{max} - \lceil \log n^3 \rceil - 2$, the total contribution to $E[OPT]$ for scales c_{min} and below is at most $OPT/2$.

The algorithm $\mathcal{A}$ simply samples a random integer c in $[c_{min}+1, c_{max}]$ and runs the greedy algorithm (choosing an element whenever doing so maintains independence), restricted to (location, element) pairs at scale c or higher. The expected contribution of this scale to OPT is $\Omega(OPT/\log n)$. It is easy to check that this gives an $O(\log n)$-competitive ratio.

References

[1] M. Babaioff, N. Immorlica, D. Kempe, and R. Kleinberg. A knapsack secretary problem with applications. In *APPROX '07*, 2007.

[2] Moshe Babaioff, Nicole Immorlica, and Robert Kleinberg. Matroids, secretary problems, and online mechanisms. In *SODA '07*, pages 434–443, 2007.

[3] Cyrus Derman, Gerald J. Lieberman, and Sheldon M. Ross. A sequential stochastic assignment problem. *Management Sci.*, 18:349–355, 1971.

[4] N. Dimitrov and C. G. Plaxton. Competitive weighted matching in transversal matroids. In *Proc. 35th Intl. Colloq. on Automata, Languages and Programming (ICALP 2008)*, 2008.

[5] E. B. Dynkin. Optimal choice of the stopping moment of a Markov process. *Dokl. Akad. Nauk SSSR*, 150:238–240, 1963.

[6] T.S. Ferguson. Who solved the secretary problem? *Statistical Science*, 4:282–296, 1989.

[7] P. R. Freeman. The secretary problem and its extensions: a review. *Internat. Statist. Rev.*, 51(2):189–206, 1983.

[8] A. Gershkov and B. Moldovanu. The dynamic assignment of heterogenous objects: A mechanism design approach. Technical report, University of Bonn, 2007.

[9] Mohammad Taghi Hajiaghayi, Robert Kleinberg, and David C. Parkes. Adaptive limited-supply online auctions. In *EC '04*, pages 71–80, 2004.

[10] Robert Kleinberg. A multiple-choice secretary algorithm with applications to online auctions. In *16th SODA*, pages 630–631. ACM, 2005.

[11] Nitish Korula and Martin Pal. Algorithms for secretary problems on graphs and hypergraphs. *arXiv:0807.1139v1 [cs.DS]*, 2008.

[12] Ron Lavi and Noam Nisan. Competitive analysis of incentive compatible on-line auctions. In *EC '00*, pages 233–241, 2000.

[13] Ron Lavi and Noam Nisan. Online ascending auctions for gradually expiring items. In *SODA '05*, pages 1146–1155, Philadelphia, PA, USA, 2005. Society for Industrial and Applied Mathematics.

[14] D.V. Lindley. Dynamic programming and decision theory. *Applied Statistics*, 10(1):39–51, March 1961.

[15] M. Mahdian, P. McAffee, and D. Pennock. The secretary problem with durable employment. personal communication, 2008.

[16] Willis T. Rasmussen and Stanley R. Pliska. Choosing the maximum from a sequence with a discount function. *Appl. Math. Optim.*, 2(3):279–289, 1975/76.

A Discounted Secretary Problems: Missing Proofs

A.1 A Sampling Lemma

The following lemma shows that given access to the element values, one can estimate the expected optimum for the discounted secretary fairly efficiently.

LEMMA A.1. *Given access to the element values and an $\epsilon > 0$, one can obtain an estimate $\widehat{Z} \in ((1-\epsilon)\mathbf{E}[\mathsf{OPT}], (1+\epsilon)\mathbf{E}[OPT])$ with probability $1-\delta$ in time* $\mathrm{poly}(n, \epsilon^{-1}, \log \delta^{-1})$.

Proof. For a parameter M to be specified later, sample m permutations $\pi_1, \pi_2, \ldots, \pi_M$ independently from $\mathfrak{S}_n$ and define

$$\widehat{Z} = \frac{1}{M}\sum_t \max_i\{d(i)v(\pi_t(i))\}.$$

Clearly, $\mathbf{E}[\widehat{Z}] = \mathbf{E}[\mathsf{OPT}]$, and it just remains to show that $\widehat{Z}$ is tightly concentrated about its mean. Let $v_{\max}$ denote the value of the most valuable element, and let $d_{\max}$ denote the largest discount. For a given trial t, let $Y_t = \frac{\max_i\{d(i)v(\pi_t(i))\}}{v_{\max}d_{\max}}$, and let $Y = \frac{1}{M}\sum_{i=1}^{M} Y_i$. (Hence $Y_i \in [0,1]$, $Y = \widehat{Z}/v_{\max}d_{\max}$.)

A quick observation: if we pick the element that appears in the location with the highest discount, the expected benefit is $d_{\max}v_{\max}/n$, and hence $\mathbf{E}[\mathsf{OPT}] \geq d_{\max}v_{\max}/n$, which in turn implies that $\mathbf{E}[Y] \geq \mathbf{E}[\widehat{Z}]/v_{\max}d_{\max} \geq 1/n$. By standard Chernoff bounds, it follows that for $\epsilon \leq 1$,

$$\begin{aligned}\mathbf{Pr}[|Y-\mathbf{E}[Y]| \geq \epsilon\mathbf{E}[Y]] &\leq 2\exp\{-\frac{\epsilon^2\cdot M\mathbf{E}[Y]}{2+\epsilon}) \\ &\leq 2\exp(-\frac{M\epsilon^2}{3n}),\end{aligned}$$

and thus if we set $M \geq \frac{3}{\epsilon^2}n\ln\frac{2}{\delta}$, this probability is at most δ, which proves the lemma.

Note that $\frac{\widehat{Z}}{1+\epsilon}$ is a lower bound for $\mathbf{E}[\mathsf{OPT}]$ with probability $1-\delta$, and hence when the values of the elements are known, we can use Theorem 4.5 to get $\frac{(1-\delta)(1-\epsilon)}{4}\mathbf{E}[\mathsf{OPT}]$.

B Reductions for Several Matroid Classes

LEMMA B.1. (GRAPHICAL MATROIDS) *Any graphical matroid satisfies a 3-partition property.*

Proof. Let G be a graph defining a graphic matroid. Pick an edge $\{u,v\}$ from G uniformly at random, with probability $1/2$ color u red and v blue, and with probability $1/2$ color u blue and v red. Then independently color every other node red or blue, each with probability $\frac{1}{2}$. Create a part in the partition matroid for each red node x, and add to it all the red-blue edges incident on x. Then run this procedure recursively on the graph induced by the edges that have both endpoints colored blue to create more sets in the partition (red-red edges are discarded).

It is easy to see that picking one bichromatic edge incident on each red node will result in a forest. It is also clear that taking the union of such a forest with any set of blue-blue edges that is itself a forest still gives a forest, implying that any set which is independent in the partition matroid we create is independent in the original graphic matroid.

For a graph G, let $\mathsf{OPT}(G)$ be the value of the optimal independent set in the graphic matroid defined by G. Let $v(G)$ be a random variable denoting the value of the maximum independent set in the partition matroid constructed by this reduction. We claim that $\mathbf{E}[v(G)] \geq \frac{1}{3}\mathsf{OPT}(G)$, and prove it by induction on the number of edges of G. For the base case, if G only has one edge then we color it bichromatically with probability 1, so $\mathbf{E}[v(G)] = \mathsf{OPT}(G) \geq \frac{1}{3}\mathsf{OPT}(G)$ as claimed.

For the inductive step, let X_{rb} be a random variable denoting the value of the optimal independent set from the partition matroid corresponding just to the sets we created for the red nodes. Let T be the optimal forest (which without loss of generality is a tree, since otherwise we just analyze each component separately). Root T arbitrarily. After the random coloring, the set of edges that go from a red node to a blue parent are clearly bichromatic and will get assigned to different sets in the partition corresponding to red nodes, so the optimal independent set in the partition matroid is at least as large as their sum. Every edge in T has probability $1/m$ (where m is the number of edges in G) of being the initial edge chosen to be colored bichromatically, and if it is the one chosen then with probability $1/2$ the parent is blue and the child is red. If it is not chosen then it still has probability $1/4$ of having its parent colored blue and its child red. So the total probability that it is colored in this way is at least $\frac{1}{2m} + (1 - \frac{1}{m})\frac{1}{4} = \frac{1}{4} + \frac{1}{4m}$, so by linearity of expectations $\mathbf{E}[X_{rb}] \geq (\frac{1}{4} + \frac{1}{4m})\mathsf{OPT}(G)$.

Clearly $v(G) = X_{rb} + v(G_{bb})$, where G_{bb} is the graph induced by edges that have both endpoints colored blue. The probability that an edge in T is colored monochromatically blue is at least $(1 - \frac{1}{m})\frac{1}{4} = \frac{1}{4} - \frac{1}{4m}$, since if it is not picked as the initial bichromatic edge its endpoints are both colored blue with probability $1/4$. By linearity of expectations this means that $\mathbf{E}[\mathsf{OPT}(G_{bb})] \geq (\frac{1}{4} - \frac{1}{4m})\mathsf{OPT}(G)$. Also, since we colored at least one edge bichromatically by induction we know that $\mathbf{E}[v(G_{bb})] \geq \frac{1}{3}\mathsf{OPT}(G_{bb})$. So $\mathbf{E}[v(G)] = \mathbf{E}[X_{rb} + v(G_{bb})] = \mathbf{E}[X_{rb}] + \mathbf{E}[v(G_{bb})] \geq (\frac{1}{4} + \frac{1}{4m})\mathsf{OPT}(G) + \mathbf{E}\left[\frac{1}{3}\mathsf{OPT}(G_{bb})\right] \geq (\frac{1}{4} + \frac{1}{4m})\mathsf{OPT}(G) + \frac{1}{3}(\frac{1}{4} - \frac{1}{4m})\mathsf{OPT}(G) \geq \frac{1}{3}\mathsf{OPT}(G)$

LEMMA B.2. (UNIFORM MATROIDS) *Any uniform matroid satisfies a $e/(e-1)$-partition property.*

Proof. Create k bins (where k is the rank of the uniform matroid), and place each element in U into one of these bins uniformly at random. The i-th largest value $v(i)$ is not in the same bin as any larger value with probability at least $(1 - 1/k)^{i-1}$. This implies that the expected value of the maximal weight basis of the partition matroid is at least

$$\begin{aligned}\textstyle\sum_{i=1}^{k} v(i) \cdot (1 - \frac{1}{k})^{i-1} &\geq \textstyle\sum_{i=1}^{k} v(i) \cdot \frac{1}{k} \sum_{j=1}^{k} (1 - \frac{1}{k})^{j-1} \\ &= \textstyle\frac{1}{k} \cdot \frac{1-(1-1/k)^k}{1-(1-1/k)} \cdot \sum_{1=1}^{n} v(i) \\ &= \textstyle(1 - (1-1/k)^k) \cdot \sum_{1=1}^{n} v(i) \\ &\geq \textstyle(1 - 1/e) \cdot \sum_{1=1}^{n} v(i)\end{aligned}$$

LEMMA B.3. (LOW-DEGREE TRANSVERSAL MATROIDS) *Transversal matroids satisfy a d-partition property, where d is the maximum degree of vertices on the left.*

Proof. Create a partition in the partition matroid for each vertex on the right, and put a vertex on the left (i.e., an element of U) in one of the bins corresponding to its neighbors uniformly at random.

Given a maximum independent set I in the transversal matroid, there is some matching between I and the vertices on the right. Clearly each node x in I gets put in the bin corresponding to its neighbor y in this matching with probability at least $1/d$. If this even does happen, then the maximum independent set in the partition matroid contains either x or an element of greater value. Thus the expected value of the maximum independent set is at least $\sum_{x \in I} \frac{1}{d} v(x) = \frac{1}{d} \sum_{x \in I} v(x) = \frac{1}{d}\mathsf{OPT}$

LEMMA B.4. (HIGH-DEGREE TRANSVERSAL MATROIDS) *Any transversal matroid satisfies a $4k/d$-partition property, where d is the minimum degree of vertices on the left and k is the rank of the matroid.*

Proof. We use the same reduction as for low-degree transversal matroids. Without loss of generality, let $v(1) \geq v(2) \geq \cdots \geq v(n)$. Then the probability that element i is the most valuable in the set that it is assigned to is at least $1 - \frac{i-1}{d}$, since at most $i - 1$ of its neighbors can contain more valuable elements and it has at least d neighbors. Considering only the elements $\{1, 2, \ldots, d/2\}$ and using linearity of expectations, we get that the expected value of the max base in the partition matroid is at least

$$\begin{aligned}\textstyle\sum_{i=1}^{d/2} \left(1 - \frac{i-1}{d}\right) v(i) &\geq \frac{1}{2}\sum_{i=1}^{d/2} v(i) \\ &\geq \frac{d}{4k}\sum_{i=1}^{k} v(i) \geq \frac{d}{4k}\mathsf{OPT}\end{aligned}$$