The Returning Secretary*

Shai Vardi[1]

1 **Blavatnik School of Computer Science, Tel Aviv University.**
Tel Aviv, Israel
shaivar1@post.tau.ac.il

Abstract

In the online random-arrival model, an algorithm receives a sequence of n requests that arrive in a random order. The algorithm is expected to make an irrevocable decision with regard to each request based only on the observed history. We consider the following natural extension of this model: each request arrives k times, and the arrival order is a random permutation of the kn arrivals; the algorithm is expected to make a decision regarding each request only upon its last arrival. We focus primarily on the case when $k = 2$, which can also be interpreted as each request arriving at, and departing from the system, at a random time.

We examine the secretary problem: the problem of selecting the best secretary when the secretaries are presented online according to a random permutation. We show that when each secretary arrives twice, we can achieve a competitive ratio of $0.767974\ldots$ (compared to $1/e$ in the classical secretary problem), and that it is optimal. We also show that without any knowledge about the number of secretaries or their arrival times, we can still hire the best secretary with probability at least $2/3$, in contrast to the impossibility of achieving a constant success probability in the classical setting.

We extend our results to the matroid secretary problem, introduced by Babaioff et al. [3], and show a simple algorithm that achieves a 2-approximation to the maximal weighted basis in the new model (for $k = 2$). We show that this approximation factor can be improved in special cases of the matroid secretary problem; in particular, we give a 16/9-competitive algorithm for the returning edge-weighted bipartite matching problem.

1998 ACM Subject Classification F.1.2 Online computation

Keywords and phrases online algorithms, secretary problem, matroid secretary

Digital Object Identifier 10.4230/LIPIcs.xxx.yyy.p

1 Introduction

The *secretary problem* [23, 10] is the following: n random items are presented to an observer in random order, with each of the $n!$ permutations being equally likely. There is complete preference order over the items, which the observer is able to query, for the items he[1] has seen so far. As each item is presented, the observer must either accept it, at which point the process ends, or reject it, and then it is lost forever. The goal of the observer is to maximize the probability that he chooses the "best" item (i.e., the one ranked first in the preference order). This problem models many scenarios; one such scenario is the one for which the problem is named: n secretaries arrive one at a time, and an interviewer must make an irrevocable decision whether to accept or reject each secretary upon arrival. Another is the

* Supported in part by the Google Europe Fellowship in Game Theory.
[1] We use male pronouns throughout this paper for simplicity. No assumption on the genders of actual agents is intended.

Conference title on which this volume is based on.
Editors: Billy Editor and Bill Editors; pp. 1–14

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

house-selling problem, in which buyers arrive and bid for the house, and the seller would like to accept the highest offer. An alternative way of modeling this problem is the following. Each secretary is allocated, independently and uniformly at random, a real number $r \in [0, 1]$, which represents his arrival time. As before, the interviewer sees the secretaries in the order of arrival and must make an irrevocable decision before seeing the next secretary. It is easy to see that the two models are essentially equivalent (assuming n is known, see e.g., [7]) - the arrival times define a permutation over the secretaries, with each permutation being equally likely. The optimal solution for the classical secretary problem is well known - wait until approximately n/e secretaries have passed[2] (alternatively until time $t = 1/e$), and thereafter, accept a secretary if and only if he is the best out of all secretaries observed so far (e.g., [15, 7]). This gives a probability of success of at least $1/e$.

Consider the following generalization of the secretary problem: Assume that each secretary arrives k times, and the interviewer has to make a decision upon each secretary's last arrival. We model this as follows: Allocate each secretary k numbers, independently and uniformly at random from $[0, 1)$, which represent his k arrival times. (Equivalently, we may consider only the order of arrivals; in this case each of the $(kn)!$ permutations over the arrival events is equally likely.) A decision whether to accept or reject a secretary must be made between his first and last arrival. We call this problem the $(k-1)$-*returning secretary problem*. The secretary problem is a classical example of the random-arrival online model (e.g., [4, 24]), and our model immediately applies to this more general framework, capturing several natural variations thereof, for example:

1. Requests may not require (or expect) an immediate answer and will therefore visit the system several times to query it.
2. When requests arrive, the system gives them either a rejection or an acceptance, or an invitation to return at some later time. It turns out that in many cases, very few requests actually need to return; in the secretary problem, for example, a straightforward analysis shows that the optimal algorithm will only ask $O(\log n)$ secretaries to return.[3]
3. Requests may enter the system and leave at some later time. The time the request stays in the system can vary from "until just before the next item arrives", in which case no information is gained, to "until the end", in which case the problem reduces to an offline one. Clearly we would like something in between. When $k = 2$, the second random variable allocated to the query can be interpreted as the time that the query leaves the system, giving a natural formulation of this property in the spirit of the random-arrival online model.

1.1 Our Results

When each secretary returns once (i.e., $k = 2$), we show that the optimal solution has a similar flavor to that of the classical secretary problem - wait until some fraction of the secretaries have passed (ignoring how many times each secretary has arrived), and thereafter hire the best secretary (out of those we have seen so far), upon his second arrival. To tightly bound the probability of success (for large n), we examine the case when each of the $2n$ arrival times is selected uniformly at random from $[0, 1)$. We use this model to show that the success probability tends to $0.767974\ldots$ as n grows. In the classical secretary problem,

[2] The exact number for each n can be computed by dynamic programming, see e.g., [15].

[3] The algorithm will only ask the i^{th} secretary that arrives to return if he is the best out of all the secretaries it has seen thus far. The probability of this is $1/i$. Summing over all secretaries gives the bound.

it is essential to know the number of secretaries arriving in order to achieve a constant success probability. We consider the case when n is not known in advance (and there is no extra knowledge, such as arrival time distribution), and show that by choosing the best secretary we have seen once he returns, with no waiting period, we can still obtain a success probability of at least 2/3. We also consider cases when $k > 2$: we show that for $k = 3$, we can achieve a success probability of at least 0.9, even without knowledge of n, and show that setting $k = \Theta(\log n)$ guarantees success with arbitrarily high probability ($1 - \frac{1}{n^\alpha}$ for any α).

We extend our results to the *matroid secretary problem*, introduced by Babaioff et al., [3], which is an adaptation of the classical secretary problem to the domain of *weighted matroids*. A weighted matroid is a pair $\mathcal{M} = (E, \mathcal{I})$ of elements E and independent sets $\mathcal{I}$, and a weight function $w : E \to \mathbb{R}$, which obeys the properties of heredity and exchange (see Section 2 for a formal definition). In the matroid secretary problem, the elements of a weighted matroid are presented in random order to the online algorithm. The algorithm maintains a set S of selected elements; when an element e arrives, the algorithm must decide whether to add it to S, under the restriction that $S \cup \{e\}$ is an independent set of the matroid. The algorithm's goal is to maximize the sum of the weights of the items in S. It is currently unknown whether there exists an algorithm that can find a set whose expected weight is a constant fraction of the optimal offline solution. The best result to date is an $O(\log \log \rho)$-competitive algorithm,[4] where ρ is the rank of the matroid, due to Lachish [22]. We show that in the returning online model, there is an algorithm which is 2-competitive in expectation (independent of the rank). We also show that for bipartite edge-weighted matching, and hence for transversal matroids[5] in general, this result can be improved, and show a 16/9-competitive algorithm.

1.2 Related Work

The origin of the secretary problem is still being debated: the problem first appeared in print in 1960 [13]; its solution is often credited to Lindley [23] or Dynkin [10]. Hundreds of papers have been published on the secretary problem and variations thereof; for a review, see [12]; for a historical discussion, see [11]. Kleinberg [19] introduced a version of the secretary problem in which we are allowed to choose k elements, with the goal of maximizing their sum. He gave a $1 - O(\sqrt{1/k})$-competitive algorithm, and showed that this setting applies to strategy-proof online auction mechanisms.

The *matroid secretary problem* was introduced by Babaioff et al. [3]. They gave an $O(\log \rho)$-competitive algorithm for general matroids, where ρ is the rank of the matroid, and several constant-competitive algorithms for special cases of the matroid secretary problem. Lachish [22] gave an $O(\log \log \rho)$ algorithm for the matroid secretary problem. There have been several improvements on special cases since then. Babaioff et al., [2] gave algorithms for the discounted and weighted secretary problems; Korula and Pál [20] showed that *graphic matroids*[6] admit $2e$-competitive algorithms; Kesselheim et al. [18] gave a $1/e$-competitive algorithm for the secretary problem on transversal matroids and showed that this is optimal.

---

[4] An online algorithm whose output is within a factor c of the optimal offline output is said to be c-competitive; see Section 2 for a formal definition.

[5] Transversal matroids (see Section 4 for a definition) are a special case of bipartite edge-weighted matching.

[6] In a *graphic matroid* $G = (V, E)$, the elements are the edges of the graph G and a set is independent if it does not contain a cycle.

Soto [28] gave a $2e^2/(e-1)$-competitive algorithm when the adversary can choose the set of weights of the elements, but the weights are assigned at random to the elements, (and the elements are presented in a random order). Gharan and Vondrák [14] showed that once the weights are random, the ordering can be made adversarial, and that this setting still admits $O(1)$-competitive algorithms. There have been other interesting results in this field; for a recent survey, see [9].

1.3 Comparison with Related Models

There are several other ther papers which consider online models with arrival and departure times. Due to the surge in interest in algorithmic game theory over the past 15 years, and the economic implications of the topic, it is unsurprising that many of these papers are economically motivated. Hajiaghayi et al. [17], consider the case of an auction in which an auctioneer has k goods to sell and the buyers arrive and depart dynamically. They notice and make use of the connection to the secretary problem to design strategy-proof mechanisms: they design an e-competititive (w.r.t. efficiency) strategy-proof mechanism for the case $k = 1$, which corresponds to the secretary problem, and extend the results to obtain $O(1)$-cometitive mechanisms for $k > 1$. Hajiaghayi et al. [16], design strategy-proof mechanisms for online scheduling in which agents bid for access to a re-usable resource such as processor time or wireless network access, and each agent is assumed to arrive and depart dynamically. Blum et al. [5], consider online auctions, in which a single commodity is bought by multiple buyers and sellers whose bids arrive and expire at different times. They present an $O(\log(p_{max} - p_{min}))$-competitive algorithm for maximizing profit and an $O(\log(p_{max}/p_{min}))$-competitive algorithm for maximizing volume where the bids are in the range $[p_{min}, p_{max}]$, and a strategy-proof algorithm for maximizing social welfare. They also show that their algorithms achieve almost optimal competitive ratios. Bredin and Parkes [6] consider online *double auctions*, which are matching problems with incentives, where agents arrive and depart dynamically. They show how to design strategy-proof mechanisms for this setting.

In Section 2 we introduce our model. In Section 3 we provide an optimal algorithm for the returning secretary problem. In the full version of the paper we give an over 2-competitive algorithm for the returning matroid secretary problem; we show that we show we can improve this competitive ratio to 16/9 for transversal matroids (and more generally, returning edge-weighted bipartite matching); and we analyze the cases of the k-returning secretary problem for $k = 3$ and $k = \Theta(\log n)$.

2 Model and Preliminaries

Consider the following scenario. There are n items which arrive in an online fashion, and each item arrives k times. Each arrival of an item is called a *round*; there are kn rounds. The order of arrivals is selected uniformly at random from the $(kn)!$ possible permutations. An algorithm observes the items as they arrive, and must make an irrevocable decision about each item upon the item's last appearance. We call such an algorithm a $(k-1)$-*returning online algorithm* and the problem it solves a $(k-1)$-*returning online problem*. Because the problem is most natural when $k = 2$, for the rest of the paper, we assume that $k = 2$ (and instead of "1-returning", we simply say "returning".) In the full version of the paper, we consider scenarios when $k > 2$.

We use the following definition of matroids:

▶ **Definition 2.1.** A matroid $\mathcal{M} = (E, \mathcal{I})$ is an ordered pair, where E is a finite set of elements (called the *ground set*), and $\mathcal{I}$ is a family of subsets of E, (called the *independent sets*), which satisfies the following properties:

1. $\emptyset \in \mathcal{I}$,
2. If $X \in \mathcal{I}$ and $Y \subseteq X$ then $Y \in \mathcal{I}$,
3. If $X, Y \in \mathcal{I}$ and $|Y| < |X|$ then there is an element $e \in X$ such that $Y \cup \{e\} \in \mathcal{I}$.

Property (2) is called the *hereditary property*. Property (3) is called the *exchange property*. An independent set that becomes dependent upon adding any element of E is called a *basis* for the matroid. In a *weighted matroid*, each element $e \in E$ is associated with a weight $w(e)$. The *returning matroid secretary problem* is the following: Each element of a weighted matroid $\mathcal{M} = (E, \mathcal{I})$ arrives twice, in an order selected uniformly at random out of the $(2n)!$ possible permutations of arrivals. The algorithm maintains a set of selected elements, S, and may add any element to S at any time between (and including) the first and second appearances of the element, as long as $S \cup \{e\} \in \mathcal{I}$. The goal of the algorithm is to maximize the sum of the weights of the elements in S. The success of the algorithm is defined by its *competitive ratio*.

▶ **Definition 2.2** (competitive ratio, c-competitive algorithm)**.** If the weight of a maximal-weight basis of a matroid is at most c times the expected weight of the set selected by an algorithm (where the expectation is over the arrival order), the algorithm is said to be *c-competitive*, and its *competitive ratio* is said to be c.

A special case of the returning matroid secretary problem is the *returning secretary problem*, in which there are n secretaries, each of whom arrives twice. The goal of the algorithm is to identify the best secretary. The algorithm is *successful* if and only if it chooses the best secretary, and we quantify how "good" the algorithm is by its success probability.

Without loss of generality, we assume throughout this paper that the weights of all the elements are distinct (this applies to secretaries as well - given any two secretaries, one must be strictly better than the other).[7] Although we do not discuss computational efficiency in this work, all the algorithms in this paper are polynomial in the succinct representation of the matroid.

We denote the set $\{1, 2, \ldots n\}$ by $[n]$.

3 The Returning Secretary

Assume that there are n secretaries that arrive in an online fashion. Each secretary arrives twice, and the order is selected uniformly at random from the $(2n)!$ possible orders. At all times, we keep note of who the best secretary is out of all the secretaries seen so far. We call this secretary the *candidate*. That is, in each round, if the secretary that arrived is better than all other secretaries that arrived before this round, he becomes the candidate. Note that it is possible that in a given round, the candidate will have already arrived twice. At any point between immediately after first arrival and immediately after the second arrival, we can *accept* or *reject* a secretary; an acceptance is final, a rejection is only final if made upon the second arrival. Once we accept a secretary, the process ends. We *win* if we accept (or *choose*) the best secretary. We would like to maximize the probability of winning.

[7] Babaioff et al., [3] show that we do not lose generality by this assumption in the matroid secretary problem. The result immediately applies to our model.

3.1 Optimal Family of Rules

What is the best strategy for maximizing the probability of winning? We first show that the optimal rule must be taken from the family of stopping rules as described in the following lemma.

▶ **Lemma 3.1.** *The optimal strategy for choosing the best secretary in the returning secretary problem has the following structure: wait until d distinct secretaries have arrived; thereafter, accept the best secretary out of the secretaries seen so far, when he returns.*

Proof. Without loss of generality, we can restrict our attention to strategies that make decisions regarding a secretary s only upon s's arrivals, as every strategy that makes decisions between the two arrival times has an equivalent strategy that defers the decision making to the second arrival. Let d_r be the random variable denoting the number of distinct secretaries that have arrived up to (and including) round r ($r \in [2n]$). Denote by $H(r) = \{x_1, x_2, \ldots, x_{r-1}\}$ the history at round r, where $x_i = (y_i, z_i)$: y_i is the relative rank (among the secretaries that have arrived until now) of the secretary that arrived at time i, and z_i represents whether this is the first or second time that this secretary has arrived (i.e. $y_i \in [d_r]$, $z_i \in \{1, 2\}$). Any (deterministic) strategy $\mathcal{S}$ must have the following structure: for every realization of $x_r = (y_r, z_r)$, and $H(r)$, $\mathcal{S}$ must accept or reject. That is $S : (H_r, y_r, z_r) \to \{\text{accept, reject}\}$. Denote the optimal strategy by $\mathcal{S}^*$. Clearly,

1. If the t^{th} secretary is not the best, we will not choose him: $\forall y_r \neq 1$, $\mathcal{S}^*(H_r, y_r, z_r) =$ reject.
2. If this is the first time we have seen a secretary, we cannot gain anything by choosing him now. It is better to wait for the second arrival, as we lose nothing by waiting: $\mathcal{S}^*(H_r, y_r, 1) =$ reject.

Therefore, we only need to consider choosing the best secretary we have seen so far when he returns; i.e., we only accept at time t such that $y_r = 1, z_r = 2$. For all other values of y_i and z_i, $\mathcal{S}^*$ must reject; henceforth, we only focus on the case that $y_r = 1, z_r = 2$, and omit this from the notation. Denote the event that $\mathcal{S}^*$ accepts on history H_r by $Acc(H_r)$. As $\mathcal{S}^*$ is a probability-maximizing strategy,

$$\mathcal{S}^*(H_r) = \text{accept} \iff \Pr[\text{win} \,|Acc(H_r)] \geq \Pr[\text{win} \,|\neg Acc(H_r)]. \tag{1}$$

Given that $d_r = d$, $\Pr[\text{win} \,|Acc(H_r)] = d/n$, as this is exactly the probability that the best secretary is part of a group of d secretaries selected uniformly at random. Although we cannot give such an elegant formula for $\Pr[\text{win} \,|\neg Acc(H_r)]$, we know that it is the probability of winning given that we have seen d secretaries, rejected them all, and have $(n-d)$ secretaries remaining to observe; hence, the probability is dependent only on d (as n is fixed). Denote this probability function by $g(d)$. We do not attempt to describe g, other than to say that g must be non-increasing in d. (This is easy to see: $g(d) \geq g(d+1)$ as a possible strategy is to always reject the d^{th} secretary.)

As the left side of (1) is an increasing function of d, and the right side is a decreasing function of d (and as $\mathcal{S}^*$ is a probability-maximizing function), $\mathcal{S}^*$ will accept only if the number of distinct secretaries that have arrived is at least d^*, the minimal d such that $d/n \geq g(d)$. We can conclude that the optimal strategy is to observe the first d^* secretaries without hiring any and to choose the first suitable secretary thereafter. It is easy to see (similarly to [8]), that randomization cannot lead to a better stopping rule. ◀

From Lemma 3.1, we can conclude that there is some function $f : n \to [0, n]$ for which the optimal algorithm for the returning secretary problem is Algorithm 1.

Algorithm 1: Returning secretary algorithm with function $f : n \to [0, n]$

Input : n, the number of secretaries
Output: A secretary s

```
the Candidate = ∅;
for round r = 1 to 2n do
    Let i_r be the secretary that arrives on round r;
    Denote by d_r the distinct number of secretaries that have arrived up to round r;
    if i_r is the Candidate then
        if d_r > f(n) then
            Return i_r;
    if i_r is better than the Candidate then
        the Candidate = i_r;
```

We do not, at this time, attempt to find the function f for which Algorithm 1 is optimized; we will optimize the parameter of a similar algorithm for a slightly different setting in Subsection 3.3. For now, we focus on the special case where $f(n) \equiv 0$, which we call the *no waiting* case. Aside from being interesting in their own right, these results will come in useful later on, for tightly bounding the success probability.

3.2 The No Waiting Case

In the classical secretary problem, even if we don't know n in advance, we can still find the best secretary with a reasonable probability, assuming we have some other information regarding the secretaries. For example, the secretaries can have an known arrival time density over $[0, 1]$ [7][8]; n can be selected from some known distribution [26]; there are other, similar scenarios (see e.g., [29, 1, 25]). However, with no advance knowledge at all, it is impossible to attain a success probability better than $1/n$ (with a deterministic algorithm): if we don't accept the first item, we run the risk of there being no other items, while if we do accept it, we have accepted the best secretary with probability $1/n$. It is easy to see that while randomization may help a little, is cannot lead to a constant success probability. In the returning-online scenario, though, we have the following result.

▶ **Theorem 3.2.** *In the returning secretary problem, even if we have no previous information on the secretaries, including the number of secretaries that will arrive, we can hire the best secretary with probability at least* $2/3$.

Denote by win the event that we hire the best secretary. Theorem 3.2 is immediate from the following lemma.

▶ **Lemma 3.3.** *When applying Algorithm 1 to the returning secretary problem with* $f(n) \equiv 0$,

$$\Pr[\text{win}] = \frac{2n+1}{3n}.$$

Proof. Let us call the best secretary Don. If we reach round i and see Don, we say we *win* on round i, and denote this event win_i. (Notice that we can say that we win at this point

[8] Note that this is different from the alternative formulation described in the introduction as in this case n is unknown.

even though this is the first time we see Don, as we will certainly hire him). The probability of winning on round 1 is exactly the probability that Don arrives first:

$$\Pr[\text{win}_1] = \frac{2}{2n}.$$

We win on round 2 if any secretary other than Don arrived on round 1, and Don arrived on round 2.

$$\Pr[\text{win}_2] = \left(\frac{2n-2}{2n}\right)\left(\frac{2}{2n-1}\right).$$

The probability of winning on round $i > 2$ is the following (the best secretary we had seen until that point did not return between rounds 2 and $i-1$, and Don arrived on round i):

$$\Pr[\text{win}_i] = \left(\frac{2n-2}{2n}\right)\left(\frac{2n-4}{2n-1}\right)\left(\frac{2n-5}{2n-2}\right)\left(\frac{2n-6}{2n-3}\right)\cdots\left(\frac{2n-i-1}{2n-i+2}\right)\left(\frac{2}{2n-i+1}\right).$$

Therefore

$$\begin{aligned}
\Pr[\text{win}] &= \frac{1}{n} + \frac{1}{n(2n-1)(2n-3)}\sum_{i=2}^{2n-2}(2n-i)(2n-i-1) \\
&= \frac{1}{n} + \frac{2(n-1)(2n-1)(2n-3)}{3n(2n-1)(2n-3)} \qquad (2) \\
&= \frac{3}{3n} + \frac{2(n-1)}{3n} \\
&= \frac{2n+1}{3n},
\end{aligned}$$

where (2) is reached by substituting $j = 2n - i$ and simplifying the sum. ◀

3.3 Optimizing the Success Probability

We would now like to optimize f in Algorithm 1 in order to maximize the algorithm's success probability. For ease of analysis, we turn to the alternative model for the secretary problem: instead of generating a random permutation over the secretaries, each secretary i is allocated, uniformly and independently at random, two real numbers $r_i^1, r_i^2 \in [0,1)$, representing his two arrival times. Assume that f^* is the optimal function for Algorithm 1. Fix n and let μ denote the time of the arrival of the $(f^*(n))^{th}$ distinct secretary. It is easy to see that the two models are asymptotically identical: for large n, $\Pr[i^j \text{is one of the first } f^*(n) \text{ arrivals}] \approxeq \Pr[i^j \in [0,\mu)]$. The analysis in this model is much cleaner, and so, for simplicity, (and at the expense of accuracy for small n), we use it to obtain our bounds. The optimal algorithm for the returning secretary problem in this model is Algorithm 2.

We introduce some new notation.

- Denote by $\text{win}(\mu)$ the event that we hire the best secretary when using Algorithm 2 with parameter μ.
- Let $\alpha_i(\mu)$ be the event that $r_i^1, r_i^2 \in [0,\mu)$.
- Let $\beta_i(\mu)$ be the event that $r_i^1 \in [0,\mu)$ and $r_i^2 \in [\mu,1)$ or vice versa.
- Let $\gamma_i(\mu)$ be the event that $r_i^1, r_i^2 \in [\mu,1)$.

We omit (μ) from the notation when it is clear from context. Label the best secretary by 1, the second best by 2 and so on. Denote by $\text{win}(NW_i)$ the event that we find the best secretary in the no waiting scenario with i secretaries (recall that this is $\frac{2i+1}{3i}$). We make the following observations, which rely on the arrival times being independent.

Algorithm 2: Returning secretary algorithm with parameter $\mu \in [0,1)$

Output: A secretary s

```
the Candidate = ∅;
Observe the first secretary;
while there are secretaries that have not arrived do
    Let i be the observed secretary;
    Let t_i be the time that i is observed;
    if i is the Candidate then
        if time ≥ μ then
            Return i;
    if i is better than the Candidate then
        the Candidate = i;
    Observe the next secretary;
```

▶ Observation 3.4. $\forall i \in [n], \Pr[\alpha_i(\mu)] = \mu^2, \Pr[\beta_i(\mu)] = 2\mu(1-\mu), \Pr[\gamma_i(\mu)] = (1-\mu)^2$.

▶ Observation 3.5. $\Pr[\text{win} \,|\gamma_1, \gamma_2, \ldots, \gamma_i, \alpha_{i+1}] = \Pr[\text{win}(NW_i)]$.

Proof. If $\gamma_1, \gamma_2, \ldots, \gamma_i$ hold then all of the appearances of the best i secretaries are in the interval $[\mu, 1)$. Both appearances of the $(i+1)^{th}$ best secretary are in $[0, \mu)$; therefore we will definitely choose one of the i best secretaries, and the probability of choosing the best is as in the no waiting scenario. ◀

▶ Observation 3.6.

$$\Pr[\text{win} \,|\gamma_1, \gamma_2, \ldots, \gamma_i, \beta_{i+1}] = \Pr[\text{win}(NW_{i+1}) | \text{secretary } i+1 \text{ is the first to arrive}].$$

Proof. If $\gamma_1, \gamma_2, \ldots, \gamma_i$ and β_{i+1} hold then all appearances of the best i secretaries are in the interval $[\mu, 1)$, and the $(i+1)^{th}$ secretary arrived once by time μ. This reduces to the problem of choosing the best secretary in the no waiting scenario, given that the $(i+1)^{th}$ secretary arrives first. ◀

▶ Claim 3.7. $\Pr[\text{win}(NW_{i+1}) | \text{secretary } i+1 \text{ is the first to arrive}] = \frac{2i}{2i+1} \Pr[\text{win}(NW_i)]$.

Proof. Given that $i+1$ is the first to arrive, if $i+1$ arrives second, we lost. If not, $i+1$ cannot be chosen anymore, and we are exactly in the no waiting scenario with i secretaries. The probability that $i+1$ arrives second given that he also arrives first is $\frac{1}{2i+1}$. ◀

Combining Observation 3.6 and Claim 3.7 gives the following corollary.

▶ **Corollary 3.8.** $\Pr[\text{win} \,|\gamma_1, \gamma_2, \ldots, \gamma_i, \beta_{i+1}] = \frac{2i}{2i+1} \Pr[\text{win}(NW_i)]$.

We are now able to obtain a recursive representation of $\Pr[\text{win} \,|\gamma_1, \gamma_2, \ldots, \gamma_i]$.

▶ Claim 3.9. $\Pr[\text{win} \,|\gamma_1, \gamma_2, \ldots, \gamma_i] = \frac{\mu^2 + 4\mu i - 2\mu^2 i}{3i} + (1-\mu)^2 \Pr[\text{win} \,|\gamma_1, \gamma_2, \ldots, \gamma_{i+1}]$.

Proof.

$$\begin{aligned}
\Pr[\text{win}\,|\gamma_1,\gamma_2,\ldots,\gamma_i] =& \Pr[\text{win}\,|\gamma_1,\gamma_2,\ldots,\gamma_i,\alpha_{i+1}]\Pr[\alpha_{i+1}]\\
&+\Pr[\text{win}\,|\gamma_1,\gamma_2,\ldots,\gamma_i,\beta_{i+1}]\Pr[\beta_{i+1}]\\
&+\Pr[\text{win}\,|\gamma_1,\gamma_2,\ldots,\gamma_i,\gamma_{i+1}]\Pr[\gamma_{i+1}]\\
=&\mu^2\Pr[\text{win}(NW_i)]+\frac{4\mu i(1-\mu)}{2i+1}\Pr[\text{win}(NW_i)] \qquad (3)\\
&+(1-\mu)^2\Pr[\text{win}\,|\gamma_1,\gamma_2,\ldots,\gamma_{i+1}]\\
=&\frac{\mu^2+4\mu i-2\mu^2 i}{2i+1}\Pr[\text{win}(NW_i)]\\
&+(1-\mu)^2\Pr[\text{win}\,|\gamma_1,\gamma_2,\ldots,\gamma_{i+1}],\\
=&\frac{\mu^2+4\mu i-2\mu^2 i}{3i}+(1-\mu)^2\Pr[\text{win}\,|\gamma_1,\gamma_2,\ldots,\gamma_{i+1}], \qquad (4)
\end{aligned}$$

where (3) is due to Observations 3.4, and 3.5 and Corollary 3.8, and (4) is due to Lemma 3.3. ◀

▶ Claim 3.10. For any constant k, and any $\mu\in[0,1)$,

$$\Pr[\text{win}]\geq 2\mu(1-\mu)+\sum_{i=1}^{k}\left(\frac{(1-\mu)^{2i}(\mu^2+4\mu i-2\mu^2 i)}{3i}\right)+\frac{2}{3}(1-\mu)^{2k+1}. \qquad (5)$$

Proof.

$$\begin{aligned}
\Pr[\text{win}] =& \Pr[\text{win}\,|\alpha_1]\cdot\Pr[\alpha_1]+\Pr[\text{win}\,|\beta_1]\cdot\Pr[\beta_1]+\Pr[\text{win}\,|\gamma_1]\cdot\Pr[\gamma_1]\\
=&0\cdot(\mu^2)+1\cdot 2\mu(1-\mu)+\Pr[\text{win}\,|\gamma_1]\cdot(1-\mu)^2, \qquad (6)
\end{aligned}$$

where (6) is due to Observation 3.4.

Recursively applying Claim 3.9, and noticing that $\Pr[\text{win}\,|\gamma_1,\gamma_2,\ldots,\gamma_i]\geq\frac{2}{3}$, for all i, completes the claim. ◀

▶ **Lemma 3.11.** *For any $x\in[0,1)$, $\Pr[\text{win}]\geq 2x-\frac{4}{3}x^2-\frac{1}{3}(1-x)^2\log(1-x^2)$.*

Proof. Substituting $x=1-\mu$ in (5), and ignoring the lowest order term, we get

$$\begin{aligned}
\Pr[\text{win}] &\geq 2x(1-x)+\frac{1}{3}\sum_{i=1}^{k}x^{2i}\left(\frac{(1-x)^2+4(1-x)i-2(1-x)^2 i}{i}\right)\\
&=2x(1-x)+\frac{1}{3}(1-x)^2\sum_{i=1}^{k}\frac{x^{2i}}{i}+\frac{1}{3}\sum_{i=1}^{k}x^{2i}(4-4x)-2(1-x)^2)\\
&=2x(1-x)+\frac{1}{3}(1-x)^2\sum_{i=1}^{k}\frac{x^{2i}}{i}+\frac{1}{3}\sum_{i=1}^{k}x^{2i}(2-2x^2)\\
&=2x(1-x)+\frac{1}{3}(1-x)^2\sum_{i=1}^{k}\frac{x^{2i}}{i}+\frac{1}{3}\left(\sum_{i=1}^{k}2x^{2i}-\sum_{i=1}^{k}2x^{2(i+1)}\right)\\
&\geq 2x(1-x)+\frac{1}{3}(1-x)^2\sum_{i=1}^{k}\frac{x^{2i}}{i}+\frac{2}{3}x^2 \qquad (7)\\
&\underset{k\to\infty}{\longrightarrow} 2x-\frac{4}{3}x^2-\frac{1}{3}(1-x)^2\log(1-x^2), \qquad (8)
\end{aligned}$$

where in (7), we once again ignore the lowest order term, and (8) is because $\sum_{i=1}^{\infty} \frac{y^i}{i}$ is the Taylor series for $-\log(1-y)$, for $|y| < 1$. ◀

Differentiating (8), we find that the winning probability is maximized at

$$x = \sqrt{\frac{e^5 - e^{W(2e^5)}}{e^{5/2}}} \approx 0.727374\ldots$$

, where $W(z)$ is the Lambert W function (also known as the the product log function). This implies $\mu \approx 0.272626\ldots$, and for this value, $\Pr[\text{win}] \approx 0.767974\ldots$. This gives our main result of the section.

▶ **Theorem 3.12.** *The success probability of Algorithm 2 with $\mu = 0.272626\ldots$ converges to the success probability of the optimal algorithm for the returning secretary problem, as n tends to infinity; the probability of hiring the best secretary using Algorithm 2 is at least* 0.767974.

4 Extension to Matroid Secretary Problems

We extend our results to the matroid secretary problem. Due to space restrictions, we only provide an outline of the results, and defer the proofs to the full version of the paper.

4.1 The Returning Matroid Secretary

We show that in the returning online model, when $k = 2$, a simple algorithm obtains a 2-approximation to the maximum-weight basis of the matroid. It is a well known property of matroids (e.g., [27]), that the Greedy algorithm always finds a maximum-weight basis. Algorithm 3, in essence, lets n elements arrive, and then runs the Greedy algorithm on the elements which have only arrived once.

▶ **Theorem 4.1.** *There is a simple algorithm for the returning matroid secretary problem that is* 2*-competitive in expectation.*

We use the following algorithm. Due to space considerations, the analysis of the algorithm and proof of Theorem 4.1 is deferred to the full version of the paper.

Algorithm 3: Returning matroid secretary algorithm

Input : a cardinality $n = |E|$ of the matroid $\mathcal{M} = (E, \mathcal{I})$
Output: an independent set $S \in \mathcal{I}$

Let n elements arrive, without choosing any element;
Let E' denote the elements which only arrived once thus far;
Relabel the elements of E' by $1, 2, \ldots, |E'|$, such that $w_1 \geq w_2 \geq \cdots \geq w_{|E'|}$;
$S \leftarrow \emptyset$;
for $i = 1$ *to* $|E'|$ **do**
  if $S \cup i \in \mathcal{I}$ **then**
    $S \leftarrow S \cup i$;
Return S;

We also show that in some cases, this algorithm can be improved to give better bounds; specifically in the case of bipartite edge-weighted matching (a generalization of transversal matroids).

4.2 Returning Bipartite Edge-Weighted Matching

The *returning bipartite edge-weighted matching* problem is a generalization of the returning transversal matroid problem.[9] Let $G = (L \cup R, E)$ be a bipartite graph with a weight function $w : E \to \mathbb{R}^+$. We are initially given R and $n = |L|$. In each step, a vertex $v \in L$ arrives together with its edges (and the edges' weights). Each vertex arrives twice, and the order of arrival is selected uniformly at random from the $(2n)!$ possible arrival orders. When a vertex $\ell \in L$ arrives for the second time, it is either matched to one of the free vertices in R that are adjacent to ℓ, or left unmatched. The goal of the algorithm is to maximize the weight of the matching. Note that if $|R| = 1$, and we succeed only if we find the maximum matching, this is exactly the returning secretary problem.

We present a variation on the returning matroid secretary algorithm, where instead of the Greedy algorithm, we use a maximum-matching algorithm (using any maximum matching algorithm, e.g., the Hungarian method [21]). We then use local improvements, similarly to [18]. Once again, we present the algorithm here, but due to space considerations, leave its analysis and the proof of Theorem 4.2 to the full version.

Algorithm 4: Returning bipartite edge-weighted matching algorithm

Input : vertex set R and a cardinality $n = |L|$
Output: a matching M

Let L_r be the vertices that arrived until round r.;
Let $L' \subset L_n$ denote the vertices that only arrived once until round n;
M = optimal matching on $G[L' \cup R]$;
for *each subsequent round* $t > n$*, when vertex* $\ell_t \in L$ *arrives* **do**
  M_t = optimal matching on $G[L_t \cup R]$;
  Let e_t be the edge assigned to ℓ_t in M_t;
  if $M \cup e_t$ *is a matching* **then**
    $M = M \cup e_t$;
Return M;

▶ **Theorem 4.2.** *Algorithm 4 is* 16/9*-competitive in expectation.*

Acknowledgements I would like to thank Yishay Mansour and the anonymous reviewers for their insightful comments.

References

1 A.R. Abdel-Hamid, J.A. Bather, and G.B. Trustrum. The secretary problem with an unknown number of candidates. *J. Appl. Prob.*, 19:619–630, 1982.
2 Moshe Babaioff, Michael Dinitz, Anupam Gupta, Nicole Immorlica, and Kunal Talwar. Secretary problems: Weights and discounts. In *SODA*, pages 1245–1254, 2009.
3 Moshe Babaioff, Nicole Immorlica, and Robert Kleinberg. Matroids, secretary problems, and online mechanisms. In *SODA*, pages 434–443, 2007.

[9] A *transversal matroid* is a bipartite graph $G = (L \cup R, E)$ where the elements are the vertices of L and the independent sets are sets of endpoints of matchings in the graph. Transversal matroids are a special case of bipartite edge-weighted matching, in which all the edges incident on the same vertex $\ell \in L$ have the same weight.

4 Bahman Bahmani, Aranyak Mehta, and Rajeev Motwani. Online graph edge-coloring in the random-order arrival model. *Theory of Computing*, 8(1):567–595, 2012.

5 Avrim Blum, Tuomas Sandholm, and Martin Zinkevich. Online algorithms for market clearing. *J. ACM*, 53(5):845–879, 2006.

6 Jonathan Bredin and David C. Parkes. Models for truthful online double auctions. *CoRR*, abs/1207.1360, 2012.

7 F. Thomas Bruss. A unified approach to a class of best choice problems with an unknown number of options. *The Annals of Probability*, 12(3):882–889, 08 1984.

8 F. Thomas Bruss. Sum the odds to one and stop. *Annals of Probability*, 28:1384–1391, 2000.

9 Michael Dinitz. Recent advances on the matroid secretary problem. *SIGACT News*, 44(2):126–142, 2013.

10 E. B. Dynkin. The optimal choice of the instant for stopping a Markov process. *Soviet Math. Dokl.*, 4:627–629, 1963.

11 Thomas S. Ferguson. Who solved the secretary problem? *Statistical Science*, 4:282–296, 1989.

12 P.R. Freeman. The secretary problem and its extensions: A review. *International Statistical Review*, 51(2):189–206, 1983.

13 M. Gardner. Mathematical games. *Scientific American*, 202:152,178–179, 1960.

14 Shayan Oveis Gharan and Jan Vondrák. On variants of the matroid secretary problem. *Algorithmica*, 67(4):472–497, 2013.

15 John P. Gilbert and Frederick Mosteller. Recognizing the maximum of a sequence. *J. Amer. Statist. Assoc.*, 61(313):35–73, 1966.

16 Mohammad Taghi Hajiaghayi, Robert D. Kleinberg, Mohammad Mahdian, and David C. Parkes. Online auctions with re-usable goods. In *EC*, pages 165–174, 2005.

17 Mohammad Taghi Hajiaghayi, Robert D. Kleinberg, and David C. Parkes. Adaptive limited-supply online auctions. In *EC*, pages 71–80, 2004.

18 Thomas Kesselheim, Klaus Radke, Andreas Tönnis, and Berthold Vöcking. An optimal online algorithm for weighted bipartite matching and extensions to combinatorial auctions. In *ESA*, pages 589–600, 2013.

19 Robert D. Kleinberg. A multiple-choice secretary algorithm with applications to online auctions. In *SODA*, pages 630–631, 2005.

20 Nitish Korula and Martin Pál. Algorithms for secretary problems on graphs and hypergraphs. In *ICALP (2)*, pages 508–520, 2009.

21 H.W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97, 1955.

22 Oded Lachish. O(log log rank) competitive ratio for the matroid secretary problem. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS*, pages 326–335, 2014.

23 D.V. Lindley. Dynamic programming and decision theory. *Appl. Statist.*, 10:39–52, 1961.

24 Mohammad Mahdian and Qiqi Yan. Online bipartite matching with random arrivals: an approach based on strongly factor-revealing LPs. In *STOC*, pages 597–606, 2011.

25 Zdzisław Porosiński. The full-information best choice problem with a random number of observations. *Stochastic Processes and their Applications*, 24(2):293–307, 1987.

26 E.L. Presman and I.M. Sonin. The best choice problem for a random number of objects. *Theory Prob. Applic.*, 17:657–668, 1982.

27 R. Rado. A note on independence functions. *Proc. London Math. Soc.*, 7:300–320, 1957.

28 José A. Soto. Matroid secretary problem in the random-assignment model. *SIAM J. Comput.*, 42(1):178–211, 2013.

29 T.J. Stewart. The secretary problem with an unknown number of options. *Operations Research*, 29:130–145, 1981.