Optimal Stopping and Applications: Lecture Notes

Vasilis Livanos*

1 Secretaries and Prophets

Optimal stopping theory concerns problems in which a decision-maker has to make a series of decisions *immediately* (upon observing new information) and *irrevocably* (without the possibility of amending a decision later on). In its most basic form, the decision-maker (henceforth *algorithm*) observes a sequence of values $x_1, \ldots, x_n \in \mathbb{R}$, where n is known a priori, that are revealed in some order. Upon observing x_i, the algorithm has to decide immediately and irrevocably whether or not to select x_i (gaining its value and ending the game), or reject it and continue to the next observation. The objective is to maximize the selected value, and the algorithm is compared to the optimal value it could have selected, i.e. $\max_i x_i$.

Within this common structure, there are many variants that one can consider. However, the two main questions that define all such models are:

1. *Who chooses the values?*
2. *Who chooses the order that they are observed in?*

On one extreme, we could consider a setting in which the answer to both questions is "an adversary". However, it is not difficult to see that in this *fully adversarial* setting, no non-trivial guarantee can be obtained by the algorithm. To motivate what follows, we show this simple fact below.

Fully Adversarial Setting. To show that no non-trivial guarantee is possible, we first need to define what constitutes an algorithm for such an online selection problem. We think of every (deterministic) algorithm as a function $f : S \to \{0, 1\}$, where $S = \left\{ \boldsymbol{x} \in \mathbb{R}^k \;\middle|\; k \leq n \right\}$. For every vector $\boldsymbol{x} = (x_1, \ldots, x_k) \in \mathbb{R}^k$ where $k \leq n$, when an algorithm $\mathcal{A}$, defined by a function $f_{\mathcal{A}}$, observes values $x_1, \ldots, x_k$ and decides to stop and select x_k, we let $f_{\mathcal{A}}(\boldsymbol{x}) = 1$. If instead the algorithm decides to reject $x_1, \ldots, x_k$ and continue, we let $f(\boldsymbol{x}) = 0$. Notice that $f_{\mathcal{A}}((x_1, \ldots, x_k)) = 1$ implies $f_{\mathcal{A}}((x_1, \ldots, x_j)) = 0$ for all $j < k$, for deterministic algorithms. Randomized algorithms are defined as convex combinations of deterministic algorithms.

To show the impossibility result, we fix an arbitrary algorithm $\mathcal{A}$ and distinguish between two cases:

- Assume that there exists a vector $\boldsymbol{x}$ with $|\boldsymbol{x}| = k < n$ such that $f_{\mathcal{A}}(\boldsymbol{x}) = 1$. In this case, the adversary can select $x_n \gg \max_{1 \leq i \leq k} x_i$. Since the algorithm obtains value x_i, we have $\frac{x_i}{x_n} \to 0$.
- Next, assume that for every vector $\boldsymbol{x}$ with $|\boldsymbol{x}| = k < n$, we have $f_{\mathcal{A}}(\boldsymbol{x}) = 0$. Thus, the only vectors $\boldsymbol{x}$ for which $f_{\mathcal{A}}(\boldsymbol{x}) = 1$ have $|\boldsymbol{x}| = n$, and the algorithm obtains a value that is at most x_n. In this case, the adversary can select a vector $\boldsymbol{x}$ with $x_1 \gg \max_{2 \leq i \leq n} x_i$. Thus, again we get $\frac{x_n}{x_1} \to 0$.

Remark 1. The impossibility result shown above holds for deterministic algorithms. However, one can easily extend it to randomized algorithms, via the following observation: if the randomized algorithm $\mathcal{A}$ is a convex combination of less than n deterministic algorithms, then for every vector of values $\boldsymbol{x}$, by the pigeonhole principle, there exists a position k such that $\Pr[f_{\mathcal{A}}(\boldsymbol{x}_{\leq k}) = 1] = 0$, where $\boldsymbol{x}_{\leq k} \in \mathbb{R}^k$ is the vector $\boldsymbol{x}$ restricted to its first k values. Similarly, if $\mathcal{A}$ is a convex combination of at least n deterministic algorithms, then for every vector of values $\boldsymbol{x}$, by the pigeonhole principle, there exists a position k such that $\Pr[f_{\mathcal{A}}(\boldsymbol{x}_{\leq k}) = 1] \leq 1/n$. In both cases, the adversary can select $x_k \gg \max_{i \neq k} x_i$, and the ratio of the algorithm's value to the optimal value again goes to 0 as n goes to infinity.

Given the above impossibility result, it is evident that, if we are to obtain any interesting guarantees, we have to relax the adversarial assumption on (at least) one of the arrival order and the values. Thus, online selection settings are split into two main categories by the choice of which assumption is relaxed: *secretary*

* These notes were assembled with a lot of help from Thanos Tolias, Marina Kontalexi and all the good people at EPFL's theory group who had the patience to sit through my lectures.

and *prophet* settings. We make the distinction between the two categories and discuss them separately due to the fact that the techniques used to show positive guarantees in each setting are inherently different, although there is sometimes some overlap between them. Having said that, a recent paper by Dughmi [**?**] shows that, at least in some settings, the two models are (roughly) equivalent.

The Secretary Problem. A first idea is to relax the assumption that the arrival order of the values is chosen by an adversary. Instead, we could assume that it is chosen at random from a (known) distribution on all $n!$ possible permutations. For simplicity, pretty much all models of this setting consider this distribution to be a uniform distribution. In this setting, we could even consider a stronger benchmark. Suppose that all x_i's are discrete and we are now satisfied only with the maximum number of the sequence. In other words, the utility of an algorithm that selects a value x_i is 0 if $x_i \neq \max_{1\leq j\leq n} x_j$ and x_i only if $x_i = \max_{1\leq j\leq n} x_j$. This is sometimes called the secretary objective. This setting where the values are chosen by an adversary (who potentially has access to our algorithm), they arrive in a uniformly random order and we are satisfied only with the maximum value in the sequence is called the *secretary problem*, and it has studied extensively in literature since its introduction by Dynkin [2] – see also Ferguson's historical discussion of the problem [5].

It is easy to see that, for the secretary objective, we cannot guarantee, in general, that we always select the maximum number; simply notice that with probability $1/n$, the largest number is the first number. Thus, we either have to always select the first number, in which case we lose with probability $1 - 1/n$, or we have to ignore the first number, in which case we lose with probability at least $1/n$. Because of this, we focus on maximizing the probability we select the maximum number, where the probability is taken over the randomness of the arrival order. The next question is what constitutes a good algorithm for this problem. This question of algorithmic design is resolved by the following two observations:

(I) First, notice that an algorithm $\mathcal{A}$ that aims to maximize $\Pr[\mathcal{A} \leftarrow \max_{1\leq i\leq n} x_i]$ should never select a value x_j if $x_j \neq \max_{1\leq i\leq j} x_i$, i.e. if x_j is not the largest value observed so far. Values that are the largest at the moment they are observed are called *records*. Thus, an optimal algorithm will always select a record.

(II) Second, notice that, by our previous observation, it is advantageous to forego early values and continue to future values, where the maximum is more likely to be, since the arrival order is chosen uniformly at random. Thus, an optimal algorithm will reject the first r values for some r and only select a value from the remaining ones.

By the above observations, an optimal algorithm will choose a position $r \in [n] = \{1, \ldots, n\}$, reject $x_1, \ldots, x_r$ and will select a record out of the remaining values $x_{r+1}, \ldots, x_n$. The first phase of the algorithm is called the *sampling phase* and the second phase is called the *selection phase*. This algorithmic scheme is formally described in Algorithm 1.

Algorithm 1: Algorithmic Scheme for the Secretary Problem $\mathcal{A}(n, r)$

```
for i ← 1 to r do
    Skip x_i
end
for i ← r + 1 to n do
    if x_i > max_{1≤j≤i−1} x_j then
        Select x_i and stop
    else
        Skip x_i
    end
end
```

To begin, let's consider the algorithm that sets $r = n/2$[1] and selects the first record out of $x_{n/2+1}, \ldots, x_n$. Interestingly, even this simple algorithm is enough to guarantee a probability of selecting the maximum value that is a constant, independent of n.

[1] We assume without loss of generality that n is even, since we can always add a dummy value $x_{n+1} = -\infty$.

Observation 1. *For every n,*

$$\Pr\left[\mathcal{A}(n, n/2) \leftarrow \max_{1\leq i\leq n} x_i\right] \geq 1/4.$$

Proof. Let $i^{(1)}$ denote the (random) position of the maximum value and $i^{(2)}$ denote the (random) position of the second highest value. Consider the two events $\mathcal{E}_1 \triangleq i^{(1)} \notin [r]$ indicating that the maximum value does not arrive in the sampling phase and $\mathcal{E}_2 \triangleq i^{(2)} \in [r]$ indicating that the second highest value arrives in the sampling phase. Clearly, $\Pr[\mathcal{E}_1] = \Pr[\mathcal{E}_2] = 1/2$, but notice that $\mathcal{E}_1$ and $\mathcal{E}_2$ are not independent. However, they are positively correlated, meaning that $\Pr[\mathcal{E}_1 \mid \mathcal{E}_2] \geq \Pr[\mathcal{E}_1]$ and $\Pr[\mathcal{E}_2 \mid \mathcal{E}_1] \geq \Pr[\mathcal{E}_2]$. Also notice that, when both $\mathcal{E}_1$ and $\mathcal{E}_2$ happen, $\mathcal{A}(n, n/2)$ will select $x_{i^{(1)}}$. Thus,

$$\Pr\left[\mathcal{A}(n, n/2) \leftarrow \max_{1\leq i\leq n} x_i\right] \geq \Pr[\mathcal{E}_1 \wedge \mathcal{E}_2] = \Pr[\mathcal{E}_1] \cdot \Pr[\mathcal{E}_2 \mid \mathcal{E}_1] \geq \Pr[\mathcal{E}_1] \cdot \Pr[\mathcal{E}_2] = \frac{1}{4}.$$

As it turns out, we can do better than $1/4$, if we optimize r. To see this, we first need to switch to a continuous time arrival model, which is equivalent to the uniformly random permutation model.

Observation 2. *Consider the set $[n]$ and suppose that each element i arrives at a time $t_i \in [0, 1]$ chosen uniformly at random and independent from all other t_j. Also, let $\boldsymbol{t}$ denote the sequence of the t_i's sorted in increasing order. Then, for any permutation π of $[n]$, we have*

$$\Pr[\boldsymbol{t} = (t_{\pi(1)}, \ldots, t_{\pi(n)})] = \frac{1}{n!}.$$

Proof. Let $r_{\boldsymbol{t}}(i) = j$ denote the index of the time t_j at position (i.e. rank) i in the sorted sequence $\boldsymbol{t}$. In other words, if t_j appears first in $\boldsymbol{t}$, we let $r_{\boldsymbol{t}}(1) = j$. Since all the t_i's are chosen uniformly at random, we have

$$\Pr\left[\bigwedge_{1\leq i\leq n} r_{\boldsymbol{t}}(i) = \pi(i)\right] = \frac{1}{n} \cdot \frac{1}{n-1} \cdot \ldots \cdot \frac{1}{1} = \frac{1}{n!}.$$

The proof follows immediately from the fact that $\Pr[\boldsymbol{t} = (t_{\pi(1)}, \ldots, t_{\pi(n)})] = \Pr\left[\bigwedge_{1\leq i\leq n} r_{\boldsymbol{t}}(i) = \pi(i)\right]$.

Given Observation 2, $\mathcal{A}(n, r)$ is equivalent to an algorithm $\mathcal{A}'(n, r/n)$ that does not select any elements of $[n]$ with arrival time in $[0, r/n]$ and selects the first record with arrival time in $(r/n, 1]$.

A natural question is whether there exists an algorithm that achieves a probability higher than $1/4$. Next, we show that the highest probability one can achieve in general is $1/e$.

Theorem 1. *For every n,*

$$\Pr\left[\mathcal{A}(n, n/e) \leftarrow \max_{1\leq i\leq n} x_i\right] \geq \frac{1}{e}.$$

Furthermore, for every $\varepsilon > 0$, there exists an $n_\varepsilon > 0$ such that, for every algorithm ALG,

$$\Pr\left[ALG \leftarrow \max_{1\leq i\leq n} x_i\right] \leq \frac{1}{e} + \varepsilon.$$

Proof. Let $i^* = \arg\max_{1\leq i\leq n} x_i$ denote the index of the maximum value. We condition on the arrival time t_{i^*} of x_{i^*} and, for an arbitrary $p \in (0, 1)$, we have

$$\Pr[\mathcal{A}'(n, p) \leftarrow x_{i^*}] = \int_p^1 \Pr[\mathcal{A}'(n, p) \leftarrow x_{i^*} \mid t_{i^*} = \tau]\, d\tau.$$

Next, given t_{i^*}, we condition again, this time on the set $S_{<t_{i^*}}$ of the indices j of values x_j that arrived before x_{i^*}. Let $x_{i^\#}$ denote the maximum value in $S_{<t_{i^*}}$, i.e. $i^\# = \arg\max_{j\in S_{<t_{i^*}}} x_j$. Clearly, $\mathcal{A}'(n, p)$ will select x_{i^*} if and only if $x_{i^\#}$ is in the sampling phase, i.e. if and only if $t_{i^\#} \leq p$. Given that $t_{i^\#} < t_{i^*}$, this happens with probability p/t_{i^*}. Thus,

$$\int_p^1 \Pr[\mathcal{A}'(n, p) \leftarrow x_{i^*} \mid t_{i^*} = \tau]\, d\tau = \int_p^1 \frac{p}{\tau}\, d\tau = p \ln\left(\frac{1}{p}\right).$$

The function $f(p) = p \ln\left(\frac{1}{p}\right)$ for $p \in (0,1]$ is maximized at $p = 1/e$, yielding $f(1/e) = 1/e$. Thus, $\Pr[\mathcal{A}(n, n/e) \leftarrow \max_{1\leq i\leq n} x_i] \geq \frac{1}{e}$.

Since every algorithm ALG has to satisfy properties (I) and (II) described above and $f(p) \leq 1/e$ for all $p \in (0,1]$, we conclude, using the equivalence shown in Observation 2, that for every $\varepsilon > 0$, there exists an $n_\varepsilon > 0$ such that $\Pr[ALG \leftarrow \max_{1\leq i\leq n} x_i] \leq \frac{1}{e} + \varepsilon$.

The Prophet Inequality. The second idea going beyond the impossibility of the fully adversarial setting is to relax the assumption that the values are chosen by an adversary. Instead, we could assume that each value x_i is a random variable X_i, drawn independently from the other values, from a (known) distribution F_i, while the arrival order is still chosen by an adversary who has full knowledge of all realizations a priori and selects the worst-case order for our algorithm. Now that each value is a random variable, what should the benchmark be? In the fully adversarial setting, the benchmark was the ratio of the value the algorithm selected over the maximum value in the sequence. Here, we take a similar approach and look at the ratio of the corresponding expected values. In other words, our benchmark is $\frac{\mathbb{E}[ALG]}{\mathbb{E}[\max_i X_i]}$. This is called the *competitive ratio* of algorithm ALG, or sometimes the prophet objective. This setting, where the arrival order of the values is chosen by an adversary, the values are drawn independently from known distributions and we want to maximize the competitive ratio is called the *prophet inequality*. The prophet inequality has a slightly shorter history than the secretary problem, having been introduced by Krengel, Sucheston and Garling [8,9], but has experienced an equally extensive study to the secretary problem in the literature – see the surveys by Correa, Foncea, Hoeksma, Oosterwijk and Vredeveld [1] and Lucier [10] for early work on the problem.

In their original paper [8], Krengel and Sucheston showed that there exists an algorithm guaranteeing a competitive ratio of $1/4$ for all instances, independent of n. Later, in collaboration with Garling [9], they presented an improved algorithm which guarantees a competitive ratio of $1/2$ for every instance. However, Krengel, Sucheston and Garling's proof is not the simplest way to guarantee a $1/2$ competitive ratio. This honour goes to Samuel-Cahn [11], who gave a simple, single-threshold algorithm with the same competitive ratio: set a threshold T equal to the median value of the distribution of $\max_i X_i$[2], and accept the first realization above T, if any. Later, Wittmann [12] showed that setting $T = \mathbb{E}[\max_i X_i]/2$ also works as a threshold. This was later independently rediscovered by Kleinberg and Weinberg [6]. Here, we present both proofs in a unified analysis[3].

Algorithm 2: Single-Threshold Algorithm for the Prophet Inequality Problem $\mathcal{B}(n, F_1, \ldots, F_n)$

```
Set T such that Pr[max_{1≤i≤n} X_i ≥ T] = 1/2 or T = E[max_{1≤i≤n} X_i]/2
for i ← 1 to n do
    if X_i ≥ T then
        Select X_i and stop
    else
        Skip X_i
    end
end
```

Theorem 2. *For every n, Algorithm 2 returns a value $\mathcal{B}(n, F_1, \ldots, F_n)$ such that*

$$\mathbb{E}[\mathcal{B}(n, F_1, \ldots, F_n)] > \frac{1}{2} \cdot \mathbb{E}[\max_{1\leq i\leq n} X_i],$$

regardless of the choice of T.

Proof. First, we upper bound $\mathbb{E}[\max_{1\leq i\leq n} X_i]$. Notice that

$$\mathbb{E}[\max_{1\leq i\leq n} X_i] \leq T + \sum_{i=1}^{n} \mathbb{E}\left[(X_i - T)^+\right],$$

[2] For simplicity, we assume that all distributions are continuous and thus a median T such that $\Pr[\max_i X_i \geq T] = 1/2$ exists.

[3] The author of these lecture notes learned of this really cool analysis that unifies both proofs from Sahil Singla.

where $(X_i - T)^+$ denotes $\max\{X_i - T, 0\}$. This actually holds for any $T \in \mathbb{R}_{\geq 0}$, since either $\max_{1\leq i\leq n} X_i \leq T$ or $\max_{1\leq i\leq n} X_i = X_j > T$ for some j, and X_j is a term that appears in the sum.

Next, we lower bound $\mathbb{E}[\mathcal{B}(n, F_1, \ldots, F_n)]$. For every $i \in [n+1]$, let $\mathcal{E}_i \triangleq \bigwedge_{j<i} (X_j < T)$ denote the event that $\mathcal{B}$ observes X_i, and notice that $\Pr[\mathcal{E}_i]$ is monotonically decreasing in i and that $\mathcal{E}_{n+1}$ is equal to the event that $\max_{1\leq i\leq n} X_i < T$. We have

$$\begin{aligned}
\mathbb{E}[\mathcal{B}(n, F_1, \ldots, F_n)] &= T \cdot \Pr\left[\max_{1\leq i\leq n} X_i \geq T\right] + \sum_{i=1}^{n} \Pr[\mathcal{E}_i] \cdot \mathbb{E}\left[(X_i - T)^+\right] \\
&\geq T \cdot \Pr\left[\max_{1\leq i\leq n} X_i \geq T\right] + \Pr[\mathcal{E}_{n+1}] \cdot \sum_{i=1}^{n} \mathbb{E}\left[(X_i - T)^+\right] \\
&= T \cdot \Pr\left[\max_{1\leq i\leq n} X_i \geq T\right] + \Pr\left[\max_{1\leq i\leq n} X_i < T\right] \cdot \mathbb{E}\left[\sum_{i=1}^{n} (X_i - T)^+\right] \\
&\geq T \cdot \Pr\left[\max_{1\leq i\leq n} X_i \geq T\right] + \Pr\left[\max_{1\leq i\leq n} X_i < T\right] \cdot \mathbb{E}\left[\max_{1\leq i\leq n} (X_i - T)^+\right] \\
&\geq T \cdot \Pr\left[\max_{1\leq i\leq n} X_i \geq T\right] + \Pr\left[\max_{1\leq i\leq n} X_i < T\right] \cdot \left(\mathbb{E}\left[\max_{1\leq i\leq n} X_i\right] - T\right).
\end{aligned}$$

Notice now that, for T such that $\Pr\left[\max_{1\leq i\leq n} X_i \geq T\right] = 1/2$, we have

$$\mathbb{E}[\mathcal{B}(n, F_1, \ldots, F_n)] \geq \frac{1}{2} \cdot T + \frac{1}{2} \cdot \left(\mathbb{E}\left[\max_{1\leq i\leq n} X_i\right] - T\right) = \frac{\mathbb{E}\left[\max_{1\leq i\leq n} X_i\right]}{2}.$$

Similarly, for $T = \mathbb{E}\left[\max_{1\leq i\leq n} X_i\right]/2$, we have

$$\begin{aligned}
\mathbb{E}[\mathcal{B}(n, F_1, \ldots, F_n)] &\geq \Pr\left[\max_{1\leq i\leq n} X_i \geq T\right] \cdot \frac{\mathbb{E}\left[\max_{1\leq i\leq n} X_i\right]}{2} \\
&\quad + \left(1 - \Pr\left[\max_{1\leq i\leq n} X_i \geq T\right]\right) \cdot \frac{\mathbb{E}\left[\max_{1\leq i\leq n} X_i\right]}{2} \\
&= \frac{\mathbb{E}\left[\max_{1\leq i\leq n} X_i\right]}{2}.
\end{aligned}$$

Next, we show that this lower bound of $1/2$ on the competitive ratio is tight, as no algorithm can guarantee a better competitive ratio.

Theorem 3. *For every $\delta > 0$, there exists a prophet inequality instance, where $n = 2$, such that, for every algorithm ALG*

$$\mathbb{E}[ALG] \leq \left(\frac{1}{2} + \delta\right) \cdot \mathbb{E}[\max\{X_1, X_2\}].$$

Proof. Consider the following instance:

$$X_1 = 1 \quad \text{w.p. } 1, \qquad X_2 = \begin{cases} 1/\varepsilon & \text{w.p. } \varepsilon \\ 0 & \text{otherwise} \end{cases}$$

We have

$$\mathbb{E}[\max\{X_1, X_2\}] = \frac{1}{\varepsilon} \cdot \varepsilon + 1 \cdot (1 - \varepsilon) = 2 - \varepsilon.$$

Every algorithm ALG observes $X_1 = 1$ and has to decide whether to select it or reject it, in which case the largest value ALG can achieve on expectation is $\mathbb{E}[X_2] = 1$. Thus, $\mathbb{E}[ALG] = 1$, and we have that $\mathbb{E}[ALG] \leq \left(\frac{1}{2} + \delta\right) \cdot \mathbb{E}[\max\{X_1, X_2\}]$, for $\delta = 1/(2 - \varepsilon) - 1/2$.

2 Combinatorial Settings

2.1 Combinatorial Feasibility Constraints

Types of Feasibility Constraints.

Matroid Primer.

2.2 The Matroid Secretary Problem

General Matroids.

Graphic Matroid Secretary. Consider the following algorithm for the secretary problem on graphic matroids, first introduced by Korula and Pal [7].

Algorithm 3: Korula-Pal Algorithm for Graphic Secretary $\mathcal{C}(G = (V, E)$

```
Let π be a uniformly random permutation of V
for e = {u, v} ∈ E do
    Orient e towards the endpoint that appears first in π
end
for u ∈ V do
    Run in parallel a single-item secretary algorithm for all e ∈ δ⁻(u)
end
```

Observation 3. *Algorithm 3 returns an acyclic set of edges F.*

Theorem 4. *Let OPT be a maximum-weight acyclic subgraph of G. Algorithm 3 returns a set of edges F such that*

$$\mathbb{E}[w(F)] \geq \frac{1}{2e} w(OPT).$$

Proof. Consider two directed graphs: graph G_0 is obtained by orienting every edge of G from higher-numbered to lower-numbered vertex in π, and graph G_1 by orienting every edge from lower-numbered to higher-numbered vertex in π.

Suppose we flip a fair coin $X \in \{0, 1\}$. For each vertex $v \in V$, let $h_X(v)$ be the heaviest edge leaving vertex v in G_X. Let $F_X = \{h_X(v) \mid v \in V\}$. Clearly, we have

$$\sum_{v \in V} w(h_0(v)) + w(h_1(v)) \geq \sum_{e \in F^*} w(e).$$

Conditioned on the coin flip X, each secretary algorithm recovers at least $1/e$ fraction of the weight of the heaviest edge leaving its vertex. Hence

$$\mathbb{E}[w(F') \mid X = x] = \frac{1}{e} w(F_x)$$

for $x = 0, 1$. By the above, we have

$$\mathbb{E}[w(F')] = \frac{1}{e} \left(\frac{1}{2}\mathbb{E}[w(F') \mid X = 0] + \frac{1}{2}\mathbb{E}[w(F') \mid X = 1] \right) \geq \frac{1}{2e} w(F^*).$$

2.3 Prophet Inequalities for Combinatorial Constraints

Matroid Prophet Inequality.

Matching Prophet Inequality. Consider the following algorithm for the prophet inequality with matching constraints, first introduced by Ezra, Feldman, Gravin and Gavin Tang [4].

Let $G = (V, E)$ be a graph and assume we have access to the probabilities $\boldsymbol{x} \in [0, 1]^{|E|}$ with which each edge e appears in the optimal (offline) solution. Next, we select a (random) set R of *active* edges, by including every edge independently with probability x_e.

Algorithm 4: A Simple Algorithm for the Matching Prophet Inequality $\mathcal{D}(G = (V, E)$

for $e = \{u, v\} \in E$ **do**
  Given the arrival order $\sigma_{\leq e}$ of the edges preceding e, calculate Pr[u, v are unmatched when e arrives]
  Define
$$\alpha_e \triangleq \frac{c}{\Pr[\text{u, v are unmatched when } e \text{ arrives}]}$$
  if *both* u, v *are unmatched* $\wedge e \in R$ **then**
    Select e with probability α_e
  end
end

Note that the term Pr[u, v are unmatched at (u, v)] involves both randomness from R and from previous steps of our algorithm.

It holds that:

$$\Pr[\text{e is matched } | e \in R] = \Pr[\text{u, v are unmatched at } (u, v)] \cdot \alpha_{(u,v)} = c.$$

It remains to show that Algorithm 4 is well-defined, i.e., that $\alpha_e \leq 1$ for all $e \in E$.

Theorem 5. *Let OPT denote the maximum-weight matching in G. Algorithm 4 returns a set F of edges such that*

$$\mathbb{E}[w(F)] \geq \frac{1}{3} E[w(OPT)].$$

Proof. Let $c = \frac{1}{3}$. We prove by induction on the number of edges that all $\alpha_e \leq 1$. The base case $|E| = 1$ is trivial, since Pr[u, v are unmatched at e] $= 1$ and $\alpha_e = \frac{1}{3}$. Let us prove the induction step. We can assume by the induction hypothesis that $\alpha_e \leq 1$ for every edge $e \in E$ but the last arriving edge (u, v). To finish the induction step, we need to show that $\alpha_e \leq 1$. Recall that our algorithm matches each edge e preceding (u, v) with probability $c \cdot x_e$. Therefore,

$$\Pr[\text{u is matched at } (u, v)] = \sum_{s \neq v} c \cdot x_{us} \leq c \quad \text{and} \quad \Pr[\text{v is matched at } (u, v)] = \sum_{s \neq u} c \cdot x_{sv} \leq c.$$

Indeed, the events that u is matched to the vertex s for each $s \in V \setminus \{v\}$ are disjoint, Pr[u matched to s] $= c \cdot x_{us}$, and $\sum_{s \neq v} x_{us} \leq 1$. Similarly, $\sum_{s \neq u} x_{sv} \leq 1$.

3 Common Variants

3.1 Matroid Secretary Variants

Random Assignment Model.

3.2 Prophet Inequality Variants

Order Variants. Given $x \in [0, 1]^n$ satisfying $\sum_i x_i \leq 1$, in this section we give a simple $(1 - 1/e)$-selectable RCRS. As a corollary, this gives an alternate proof of the $(1 - 1/e)$-prophet secretary for a single item due to Esfandiari et al. [3].

We first notice that the random order can be emulated by assuming each element i selects a random time t_i to arrive uniformly at random in the interval $[0, 1]$.

Theorem 6. *An algorithm that selects an active element i arriving at time $t \in [0, 1]$ with probability $\exp(-t \cdot x_i)$ (and ignores i otherwise) is $(1 - 1/e)$-selectable for a rank 1 matroid; that is, on average, this algorithm considers (does not ignore) any element i at least $(1 - 1/e)$ fraction of the time.*

Proof. By reaching time t (element j), let us denote the event that no element is selected before time t (element j's arrival). We start by noticing that for any element $i \in [n]$,

$$\Pr[i \text{ is considered}] = \int_0^1 \Pr[i \text{ is considered at time } t \mid \text{reach time } t \text{ and } i \text{ arrives at } t] \cdot \Pr[\text{reach time } t \mid i \text{ arrives at } t]\, dt.$$

This simplifies to

$$\Pr[i \text{ is considered}] = \int_0^1 \exp(-t \cdot x_i) \cdot \Pr[\text{reach time } t \mid i \text{ arrives at } t]\, dt.$$

Now, we can simplify

$$\Pr[\text{reach time } t \mid i \text{ arrives at } t] = \prod_{j \neq i} \left(1 - \Pr[j \text{ arrives before } t \text{ and is active and considered} \mid \text{reach } j]\right).$$

This equals

$$\prod_{j \neq i} \left(1 - x_j \cdot \int_0^t \exp(-a \cdot x_j)\, da\right) = \prod_{j \neq i} \exp(-t \cdot x_j).$$

Thus, combining with the previous equation,

$$\Pr[i \text{ is considered}] = \int_0^1 \exp(-t \cdot x_i) \cdot \prod_{j \neq i} \exp(-t \cdot x_j)\, dt \geq \int_0^1 \exp(-t)\, dt = 1 - \frac{1}{e},$$

where the inequality uses $\sum_i x_i \leq 1$.

- Prophet Secretary
- Free Order
- IID (do the DP)

$$\mathop{\mathbb{E}}_{X_1,\ldots,X_n}[ALG(X_1,\ldots,X_n)] = \mathop{\mathbb{E}}_{X_1}\left[\max\left\{X_1, \mathop{\mathbb{E}}_{X_2,\ldots,X_n}[ALG(X_2,\ldots,X_n)]\right\}\right].$$

Information Variants.

- Sample Access
- Limited Distributional Knowledge
- Oracle Access

4 Techniques

4.1 Secretary Techniques

Sampling & Selection

Greedy Improving.

α-Partition Property.

Matroid Decomposition.

4.2 Prophet Inequality Techniques

Balanced Prices.

Online Contention Resolution Schemes. (Due to Chawla, Hartline, Malec, and Sivan; Alaei.) Define p_i as the probability that element X_i takes on the highest value. Hence, $\sum_i p_i = 1$. Moreover, suppose ξ_i is such that $\Pr[X_i \geq \xi_i] = p_i$, i.e., the p_i-th percentile for X_i. Define

$$v_i(p_i) := \mathbb{E}[X_i \mid X_i \geq \xi_i]$$

as the value of X_i conditioned on it lying in the top p_i-th percentile. Clearly, $\mathbb{E}[X_{\max}] \leq \sum_i v_i(p_i) \cdot p_i$.

Here's a simple algorithm that gets value $\frac{1}{4}\sum_i v_i(p_i) \cdot p_i \geq \frac{1}{4}\mathbb{E}[X_{\max}]$. If we have not chosen an item among $1, \ldots, i-1$, when looking at item i, pass with probability $\frac{1}{2}$ without even looking at X_i, else accept it if $X_i \geq \xi_i$.

Say we "reach" item i if we've not picked an item before i. The expected value of the algorithm is

$$\text{ALG} \geq \sum_{i=1}^{n} \Pr[\text{reach item } i] \cdot \frac{1}{2} \cdot \Pr[X_i \geq \xi_i] \cdot \mathbb{E}[X_i \mid X_i \geq \xi_i] = \sum_{i=1}^{n} \Pr[\text{reach item } i] \cdot \frac{1}{2} \cdot p_i \cdot v_i(p_i). \quad (1.1)$$

Since we pick each item with probability $\frac{1}{2}p_i$, the expected number of items we choose is half. So, by Markov's inequality, the probability we pick no item at all is at least half. Hence, the probability we reach item i is at least one half too, and the above expression is $\frac{1}{4}\sum_i v_i(p_i) \cdot p_i$ as claimed.

To improve this algorithm, suppose we denote the probability of reaching item i by r_i, and suppose we reject item i outright with probability $1 - q_i$. Then (1.1) really shows that

$$\text{ALG} \geq \sum_{i=1}^{n} r_i \cdot q_i \cdot p_i \cdot v_i(p_i).$$

Above, we ensured that $q_i = r_i = \frac{1}{2}$, and hence lost a factor of $\frac{1}{4}$. But if we could ensure that $r_i \cdot q_i = \frac{1}{2}$, we'd get the desired result of $\frac{1}{2}\mathbb{E}[X_{\max}]$. For the first item $r_1 = 1$ and hence we can set $q_1 = \frac{1}{2}$.

What about future items? Note that since

$$r_{i+1} = r_i(1 - q_i \cdot p_i), \quad (1.2)$$

we can just set $q_{i+1} = \frac{1}{2r_{i+1}}$. A simple induction shows that $r_{i+1} \geq \frac{1}{2}$—indeed, summing up (1.2) gives

$$r_{i+1} = r_1 - \frac{1}{2}\sum_{j \leq i} p_j,$$

so that $q_{i+1} \in [0, 1]$ and is well-defined.

The Power of the Adversary.

Greedy Online Contention Resolution Schemes. The following scheme is due to Jan Vondrák. Let π denote the OCRS we will create. π will draw a random set R where each element e_i appears in R independently with some probability q_i. Afterwards, it will set

$$F_{\pi,x} = \{\{e_i\} \mid e_i \in R\}.$$

We set $q_i = \frac{1-e^{-x_i}}{x_i}$ for all $e_i \in N$. Afterwards, π selects the first element e_i that is active and that $\{e_i\} \in F$.

The proof of the next lemma is identical to the proof of Lemma 3.2.

Lemma 1 (Lemma A.3). *π is a randomized greedy OCRS.*

Next, we quantify the probability that each element is selected by π, given that it is active.

Lemma 2 (Lemma A.4). *π selects every element $e_i \in N$, given that it is active, with probability at least $1/e$.*

Proof. We relabel the elements of N so that each e_i arrives in the i-th step. Consider an element $e_i \in N$. Given that e_i is active, since π is a greedy OCRS, π will select e_i if and only if it has not selected any elements before e_i and also $\{e_i\} \in F_{\pi,x}$. Recall that we have $\{e_i\} \in F_{\pi,x}$ with probability exactly $q_i = \frac{1-e^{-x_i}}{x_i}$. Furthermore, for every element e_j where $j < i$, it needs to be the case that we avoid having both $\{e_j\} \in F_{\pi,x}$ and also e_j coming up active. This happens with probability $1 - x_j \cdot \frac{1-e^{-x_j}}{x_j} = e^{-x_j}$ for every e_j where $j < i$.

Overall, if we denote by r_i the probability that e_i is selected by π, given that it is active, we have

$$r_i = \frac{1-e^{-x_i}}{x_i} \cdot \prod_{j<i} e^{-x_j} = \frac{1-e^{-x_i}}{x_i} \cdot e^{-\sum_{j<i} x_j} \geq \frac{e^{x_i-1} - e^{-1}}{x_i},$$

where the inequality follows from $\sum_i x_i \leq 1$. This expression is minimized for $x_i \to 0$, and thus we get $r_i \geq \frac{1}{e}$, for all $i \in N$.

From Lemmas A.3 and A.4, it follows that π is a $1/e$-selectable (randomized) greedy OCRS for P.

5 Applications

- Auctions
- Stochastic Optimization
  - Online Matching (e.g. Tasks to Agents)
  - Vehicle Routing and Assignment
- Dynamic Resource Allocation
  - Cloud Computing and Resource Allocation
  - Network Bandwidth Allocation
- Supply Chain and Inventory Management
  - Inventory Restocking
  - Order Fulfillment
- Portfolio Selection and Financial Decision Making
- Organ Transplant Allocation
- Exploration vs. Exploitation in Bandit Problems
- Online Job Scheduling with Uncertain Durations

5.1 Single-Item Auction

Second-price auction – maximize Social Welfare.

5.2 (Offline) Combinatorial Auctions.

The Complement-Free Hierarchy. Here it is:

1. Additive/Linear/Modular
2. Gross Substitutes
3. Submodular
4. XOS/Fractionally Subadditive
5. Subadditive
6. General Monotone

Definition 1 (Gross Substitutes (GS)). *The original GS definition is based on a price vector and a demand set.*

- *A price vector p is a vector containing a price for each item.*
- *Given a utility function u and a price vector p, a set X is called a demand if it maximizes the net utility of the agent:*

$$u(X) - p \cdot X.$$

– *The demand set $D(u,p)$ is the set of all demands.*

The GS property means that when the price of some items increases, the demand for other items does not decrease. Formally, for any two price vectors q and p such that $q \geq p$, and any $X \in D(u,p)$, there is a $Y \in D(u,q)$ such that

$$Y \supseteq \{x \in X \mid p_x = q_x\}$$

(Y contains all items in X whose price remained constant).

Oracle Models Definitions

The Vickrey-Clarke-Groves (VCG) Mechanism. Externalities.
However, it's NP-Hard. We want polynomial-time approximations.

Results for (Offline) Combinatorial Auctions. If v_i's have polynomial size representations and we can exactly maximize welfare, we can use the VCG mechanism (in polynomial time) to incentivize truthful reporting of valuations. If valuations do not have polynomial representations but we can use queries in order to maximize welfare (which is the case for GS, we note that answering demand queries for GS valuations can be done in polynomial time), then the implementation is not in dominant strategies, but is incentive compatible in some weaker sense.

LP for (Offline) Combinatorial Auction with Unit Demand Valuations ≡ Maximum-Weight Matching

$$\begin{aligned}
\max \quad & \sum_{i \in N} \sum_{j \in M} v_{ij} x_{ij} \\
\text{s.t.} \quad & \sum_{i \in N} x_{ij} \leq 1, \quad \forall j \in M \\
& \sum_{j \in M} x_{ij} \leq 1, \quad \forall i \in N \\
& x_{ij} \geq 0, \qquad \forall i \in N, j \in M
\end{aligned}$$

– Value Oracle:
  - Unit Demand: 1-approximation. BNIC.
  - GS: 1-approximation. BNIC.
  - Submodular: $1 - 1/e$-approximation, tight. Not IC.
  - XOS: $\mathrm{O}\left(1/\sqrt{m}\right)$-approximation, tight. Not IC.
  - Subadditive: $\mathrm{O}\left(1/\sqrt{m}\right)$-approximation, tight. Not IC.
– Demand Oracle:
  - Unit Demand: 1-approximation. BNIC.
  - GS: 1-approximation. BNIC.
  - Submodular: δ-approximation for some $1 - 1/e < \delta < 0.938$. Not IC. One can easily get a $1 - 1/e$-approximation using a $1 - 1/e$-selectable (offline) CRS.
  - XOS: $1 - 1/e$-approximation, tight. Not IC.
  - Subadditive: $1/2$-approximation, tight. Not IC.

Techniques. Configuration LP: One variable per $i \in N, S \subseteq M$.

$$\begin{aligned}
\max \quad & \sum_{i \in N} \sum_{S \subseteq M} v_i(S) x_{i,S} \\
\text{s.t.} \quad & \sum_{S \subseteq M} x_{i,S} \leq 1, \qquad \forall i \in N \\
& \sum_{S \subseteq M : j \in M} x_{i,S} \leq 1, \forall j \in M \\
& x_{i,S} \geq 0, \qquad \forall i \in N, S \subseteq M
\end{aligned}$$

Reduction of Submodular Combinatorial Auctions to Submodular Function Maximization:

Let the set of n players be P, the set of m items Q, and for each $i \in P$, let the respective utility function be $w_i : 2^Q \to \mathbb{R}^+$. We define a new ground set $X = P \times Q$, with a function $f : 2^X \to \mathbb{R}^+$ defined as follows: Every set $S \subseteq X$ can be written uniquely as $S = \bigcup_{i \in P}(\{i\} \times S_i)$. Then let

$$f(S) = \sum_{i \in P} w_i(S_i).$$

Assuming that each w_i is a monotone submodular function, it is easy to verify that f is also monotone submodular. The interpretation of this construction is that we make $|P|$ copies of each item, one for each player. However, in reality we can only allocate one copy of each item. Therefore, let us define a partition matroid $M = (X, I)$ as follows:

$$I = \{S \subseteq X \mid \forall j; |S \cap (P \times \{j\})| \leq 1\}.$$

Then the Submodular Combinatorial Auction is equivalent to

$$\max\{f(S) : S \in I\},$$

which one can solve and get a $1 - 1/e$-approximation, for example via continuous greedy.

Truthfulness

- The stronger degree is *dominant-strategy incentive-compatibility (DSIC)*. It means that truth-telling is a weakly-dominant strategy, i.e. you fare best or at least not worse by being truthful, regardless of what the others do. In a DSIC mechanism, strategic considerations cannot help any agent achieve better outcomes than the truth; such mechanisms are called strategyproof, truthful, or straightforward.
- A weaker degree is *Bayesian-Nash incentive-compatibility (BNIC)*. It means there is a Bayesian Nash equilibrium in which all participants reveal their true preferences. In other words, if all other players act truthfully, then it is best to be truthful.

VCG is the only DSIC optimal mechanism. Results for Truthful (Offline) Combinatorial Auctions.

- Value Oracle:
  - GS: 1-approximation. BNIC.
  - Submodular: $m^{\Omega(1)}$-approximation, tight. Not IC.
  - XOS: $m^{\Omega(1)}$-approximation, tight. Not IC.
  - Subadditive: $m^{\Omega(1)}$-approximation, tight. DSIC.
- Demand Oracle:
  - GS: 1-approximation. BNIC.
  - Submodular: $(\log \log m)^2$-approximation, not tight. DSIC. [AKS'21]
  - XOS: $(\log \log m)^2$-approximation, not tight. DSIC. [AKS'21]
  - Subadditive: $(\log \log m)^3$-approximation, not tight. DSIC. [AKS'21]

Problems with VCG Here are some:

1. NP-Hard to compute allocation
2. Revenue Guarantees
3. Collusion

Example 1. Consider an auction with two items, s_1 and s_2. Suppose we have collected two bids (from different bidders), both (s_1, s_2, N). If these are the only two bids, one of the bidders will be awarded both the items and, under the VCG mechanism, will have to pay N.

However, suppose two more bids (by different bidders) come in: $(s_1, N+1)$ and $(s_2, N+1)$. Then these bids will win. Moreover, neither winning bidder will have to pay anything! (This is because a winning bidder's item would simply be thrown away if that winning bidder were removed.)

This example demonstrates a number of issues:

1. The addition of more bidders can actually decrease the auctioneer's revenue from an arbitrary amount to 0.
2. The VCG mechanism is not revenue-equivalent to the sealed-bid first-price mechanism in combinatorial auctions, even when all bidders' true valuations are common knowledge[4].

[4] Unlike in the single-item case.

3. Even when the other bidders by themselves would generate non-negative revenue for the auctioneer under the VCG mechanism, it is possible that two colluding bidders can bid so as to receive all the items without paying anything.

5.3 Online Combinatorial Auctions.

Suppose now that the agents cannot lie, but their valuations v_i are random variables, drawn from known distributions F_i over valuations. Suppose also that they arrive in an online fashion and when buyer i arrives, we get to observe all of $v_i(S)$ for all sets S, simultaneously. What can we do?

Posted-Price Mechanisms – equivalence with prophet inequalities. Trivial strategyproof-ness.

- GS: 1/2-approximation. BNIC.
- Submodular: 1/2-approximation, not tight. DSIC. [AKS'21]
- XOS: 1/2-approximation, not tight. DSIC. [AKS'21]
- Subadditive: 1/6-approximation, but without pricing, via a fixed-point argument.

5.4 Extensions

Revenue Maximization. What about maximizing revenue? For single-item auctions, use Myerson's optimal auction: maximizing *virtual welfare* $\phi_i(v) = v - \frac{1-F_i(v)}{f_i(v)}$ maximizes revenue. $\phi_i(v)$ can be negative, in which case it's optimal to not allocate the item!

Subject to selling the item, VCG is revenue-optimal.

What about simple mechanisms?

Combinatorial Auctions with Feasibility Constraints. Matchings, Matroids and even general feasibility constraints ($\mathrm{O}\left(\log n \log^2 r\right)$ for online CAs, due to [RS'17]).

Prior-Free Auctions. Connections with secretary-type problems.

Sample Access.

Communication Complexity.

6 Conclusion and Open Problems

6.1 Further Work

Benchmark Variants – Philosopher's Problem.

Learning-Augmented Settings – ties with ML.

6.2 Open Problems

Matroid Secretary.

Free Order vs IID.

Prophet Secretary.

$1/e$**-selectable Greedy OCRS for Matroids.**

Optimal Submodular CAs.

$\mathrm{O}\,(1)$**-factor Truthful CAs.**

Online Subadditive Auctions with Prices.

Correlated Prophet Inequalities & OCRSs.

References

1. Correa, J.R., Foncea, P., Hoeksma, R., Oosterwijk, T., Vredeveld, T.: Recent developments in prophet inequalities. SIGecom Exch. **17**(1), 61–70 (2018). https://doi.org/10.1145/3331033.3331039, https://doi.org/10.1145/3331033.3331039
2. Dynkin, E.B.: The optimum choice of the instant for stopping a Markov process. Soviet Math. Dokl **4** (1963)
3. Esfandiari, H., Hajiaghayi, M., Liaghat, V., Monemizadeh, M.: Prophet secretary. SIAM Journal on Discrete Mathematics **31**(3), 1685–1701 (2017). https://doi.org/10.1137/15M1029394, https://doi.org/10.1137/15M1029394
4. Ezra, T., Feldman, M., Gravin, N., Tang, Z.G.: Prophet matching with general arrivals. Mathematics of Operations Research **47**(2), 878–898 (2022). https://doi.org/10.1287/moor.2021.1152, https://doi.org/10.1287/moor.2021.1152
5. Ferguson, T.S.: Who Solved the Secretary Problem? Statistical Science **4**(3), 282 – 289 (1989). https://doi.org/10.1214/ss/1177012493, https://doi.org/10.1214/ss/1177012493
6. Kleinberg, R., Weinberg, S.M.: Matroid prophet inequalities and applications to multi-dimensional mechanism design. Games Econ. Behav. **113**, 97–115 (2019). https://doi.org/10.1016/j.geb.2014.11.002, https://doi.org/10.1016/j.geb.2014.11.002
7. Korula, N., Pál, M.: Algorithms for secretary problems on graphs and hypergraphs. In: Albers, S., Marchetti-Spaccamela, A., Matias, Y., Nikoletseas, S., Thomas, W. (eds.) Automata, Languages and Programming. pp. 508–520. Springer Berlin Heidelberg, Berlin, Heidelberg (2009)
8. Krengel, U., Sucheston, L.: Semiamarts and finite values. Bull. Amer. Math. Soc. **83**(4), 745–747 (07 1977), https://projecteuclid.org:443/euclid.bams/1183538915
9. Krengel, U., Sucheston, L.: On semiamarts, amarts, and processes with finite value. Probability on Banach spaces **4**, 197–266 (1978)
10. Lucier, B.: An economic view of prophet inequalities. SIGecom Exch. **16**(1), 24–47 (Sep 2017). https://doi.org/10.1145/3144722.3144725, https://doi.org/10.1145/3144722.3144725
11. Samuel-Cahn, E.: Comparison of threshold stop rules and maximum for independent nonnegative random variables. The Annals of Probability **12**(4), 1213–1216 (1984), http://www.jstor.org/stable/2243359
12. Wittmann, R.: Prophet inequalities for dependent random variables. Stochastics and Stochastic Reports **52**(3-4), 283–293 (1995). https://doi.org/10.1080/17442509508833976, https://doi.org/10.1080/17442509508833976