CS 466/666: Algorithm Design and Analysis **University of Waterloo: Fall 2023**

Lecture 13

October 25, 2023

Instructor: Sepehr Assadi

Topics of this Lecture

1 Online Decision Making

We continue the study of online algorithms, focusing on a canonical scenario: how to find the best option among many options presented to us in an online fashion? To motivate this, suppose we have a set of n candidates for a job and our goal is to hire the best one. The twist is that we get to see these candidates one by one and we have to decide right away whether we want to hire this candidate or pass on this opportunity and try the remaining ones. The important part is that these decisions are irrevocable: if we hire a candidate, we can no longer examine the remaining ones and thus we may lose on much better candidates coming in the future; conversely, if we pass on a candidate, we may just lose our chance of hiring the best option and have to settle for worse candidates in the future.

These problems are at the heart of various online decision making processes and have been studied in probability theory, mathematics, theoretical computer science, and optimization literature for over half a decade. In this lecture, we consider two of the most canonical formalizations of these problems:

- **The secretary problem**: wherein we have no knowledge about the "absolute values" of the candidates and instead can only compare their *relative values* with each other, i.e., decide if the incoming one is better or worse, then a previously seen candidate. In this setting, we assume that the candidates are coming in a *random order* and the goal is to **maximize the probability** of finding the best candidate.
- **The prophet inequality**: wherein we know a distribution for the value of each candidate but do not know the actual value of the candidate sampled from their distribution before their arrival. In this setting, the order of arrival of candidates is chosen *adversarially* but the distributions are independent of each other; the goal is to **maximize the expected value** of the chosen candidate compared to a "prophet" that can see the actual value of all the candidates and can simply pick the best one.

We formalize and study each of these problems in the following two sections.

2 The Secretary Problem

Formally, the secretary problem is defined as follows. We have a set of n elements $a_1, \ldots, a_n$ with a total ordering between them. We are given a uniformly random permutation of these elements, denoted by

$a_{[1]}, \ldots, a_{[n]}$. At each time step $t \in [n]$, we see the element $a_{[t]}$ and can only compare it with the previously seen elements $a_{[1]}, \ldots, a_{[t-1]}$. Moreover, we can either *choose* this element and finish the process, or *pass on* this element and see the next one. The goal is to find a strategy ALG that maximizes the probability of choosing the best element, namely, the largest element among $a_1, \ldots, a_n$ according to the total ordering. For simplicity, we we say ALG 'wins' if it picks the best element.

Warm-up: A simple strategy. Let us start be examining a very simple strategy. We pass on the first $(n/2)$ arriving elements completely. Then, in the remaining $(n/2)$ elements to arrive, we pick the first one that is better than all the ones seen so far. How good this algorithm does?

Lemma 1. *The above simple strategy outputs the best element with probability at least* $1/4$.

Proof. Consider the following two events:

- E_1: the second best element appears in the first half;
- E_2: the best element appears in the second half.

Notice that if both events E_1 and E_2 happens, then the strategy wins: this is because the "threshold" set by the first half is the second best element and thus the only element in the second half that is larger than this threshold is the best element. We thus have,

$$\begin{aligned}
\Pr(\text{win}) &\geqslant \Pr(E_1 \wedge E_2) && \text{(as discussed above)} \\
&= \Pr(E_1) \cdot \Pr(E_2 \mid E_1) \\
&= \frac{n/2}{n} \cdot \Pr(E_2 \mid E_1) \\
& \text{(the second best element is placed in a random position and } E_1 \text{ happens if it is in the first half)} \\
&= \frac{n/2}{n} \cdot \frac{n/2}{n-1} \\
& \text{(the best element is placed randomly among remaining } n-1 \text{ positions conditioned on } E_1\text{)} \\
&= \frac{1}{4} + o(1).
\end{aligned}$$

□

Lemma 1 already provides a surprising answer: no matter the number of elements n, the probability of winning is an absolute constant! But, we can do even better than this particular constant of $1/4$. A simple glance at the proof of Lemma 1 reveals that the only real place that we had a "true" inequality is in the first line, namely, replacing the event 'win' with that of $E_1 \wedge E_2$. To see why this is a loose inequality, consider the following event: the third best element appears in the first half, and the best element appears in the second half before the second-best element. In this case also, the algorithm wins even though the event E_1 has not happened actually. We will see how to use this fact to obtain a better bound.

A better strategy. Let us show a better strategy for this problem which in fact is *optimal* although we are not going to prove its optimality here (it is a relatively easy exercise to prove that, so try it on your own!). The strategy is almost exactly as before with a simple change. We will pick an integer $k \in [n]$ to be determined later and will partition the arriving elements to the first k elements and the remaining $n-k$ elements. We again pass on all the elements in the first part and then pick the first element in the second half that is larger than all the ones seen so far (so, the basic algorithm is the same as this one for $k = n/2$).

Lemma 2. *The better algorithm described above outputs the best element with probability at least*

$$(1 - o(1)) \cdot \frac{k}{n} \cdot \ln\left(\frac{n}{k}\right),$$

as long as $n, k = \omega(1)$.

Proof. Consider an index $i > k$ and suppose $a_{[i]}$ is the best element; how can the algorithm win in this case, i.e., chooses element $a_{[i]}$? For that to happen, we need that the algorithm does not stop on $a_{[1]}, \ldots, a_{[i-1]}$ and for that to happen, we need the largest element in this set to be among the first k elements (if not, the algorithm chooses that element instead and terminate before reaching $a_{[i]}$) – this is an if and only if case since if the algorithm reaches $a_{[i]}$, it will certainly choose it. We can thus write,

$$
\begin{aligned}
\Pr(\text{win}) &= \sum_{i=k+1}^{n} \Pr\left(a_{[i]} \text{ is the best element} \wedge \text{the algorithm chooses } a_{[i]}\right) \\
&\qquad \text{(the algorithm chooses at most one element, so the events in the sum are disjoint and we can add them)} \\
&= \sum_{i=k+1}^{n} \Pr\left(a_{[i]} \text{ is the best element} \wedge \text{best of } a_{[1]}, \ldots, a_{[i-1]} \text{ appears in first } k \text{ elements}\right) \\
&\qquad \text{(by the discussion above)} \\
&= \sum_{i=k+1}^{n} \Pr\left(a_{[i]} \text{ is the best element}\right) \cdot \Pr\left(\text{best of } a_{[1]}, \ldots, a_{[i-1]} \text{ appears in first } k \text{ elements}\right) \\
&\qquad \text{(the two events are independent as the second one is only a function of relative order of first } i-1 \text{ elements)} \\
&= \sum_{i=k+1}^{n} \frac{1}{n} \cdot \frac{k}{i-1} \\
&\qquad \text{(best element is placed randomly in } [n]\text{, and best of first } i-1 \text{ is placed randomly in } [i-1]) \\
&= \frac{k}{n} \cdot \left(\frac{1}{k} + \frac{1}{k+1} + \cdots + \frac{1}{n-1}\right) \\
&= \frac{k}{n} \cdot \left(1 + \frac{1}{2} + \cdots + \frac{1}{n-1} - (1 + \frac{1}{2} + \cdots + \frac{1}{k-1})\right) \\
&\qquad \text{(by adding and subtracting } 1 + 1/2 + \cdots + 1/(k-1)) \\
&= \frac{k}{n} \cdot (H_{n-1} - H_{k-1}) \qquad \text{(where } H_j \text{ is the } j\text{-th Harmonic number)} \\
&\geqslant (1 - o(1)) \cdot (\ln(n-1) - \ln(k-1)) \qquad \text{(as } H_j \approx \ln j \text{ for } j = \omega(1)) \\
&\geqslant (1 - o(1)) \cdot \ln\left(\frac{n}{k}\right). \qquad \text{(as } \ln(n-1) - \ln(k-1) = \ln\left(\tfrac{n-1}{k-1}\right) \approx \ln\left(\tfrac{n}{k}\right) \text{ for } n, k = \omega(1))
\end{aligned}
$$

This concludes the proof. □

Our task is now to simply find the best value of k that maximizes the quantity in Lemma 2 (it is easy to see that k should be $\Theta(n)$ and thus we do not need to worry about the condition $k = \omega(1)$). Define the function

$$
f(x) := \frac{x}{n} \cdot \ln\left(\frac{n}{x}\right) = \frac{1}{n} \cdot (x \ln n - x \ln x).
$$

Our goal is to maximize x over the interval $x \in (0, n)$. The first and second derivative of f are:

$$
\begin{aligned}
f'(x) &= \frac{1}{n} \cdot \left(\ln n - \ln x - x \cdot \frac{1}{x}\right) = \frac{1}{n} \cdot \left(\ln\left(\frac{n}{x}\right) - 1\right) \\
f''(x) &= \frac{1}{n} \cdot \left(-\frac{1}{x}\right).
\end{aligned}
$$

So, in the interval $(0, 1)$, f is a concave function and thus its maximum happens either on the boundaries (which is not the case for this f) or when $f'(x) = 0$, which gives us

$$
\ln\left(\frac{n}{x}\right) = 1 \implies x = \frac{n}{e}.
$$

Thus, the best choice for the algorithm above is to set $k = n/e$ and achieve

$$
\Pr(\text{win when } k = n/e) \geqslant (1 - o(1)) \cdot \frac{k}{n} \cdot \ln\left(\frac{n}{k}\right) \approx \frac{1}{e}.
$$

To conclude, there is a strategy for the secretary problem that achieves a probability of success of $\approx 1/e$. As stated earlier, this algorithm is also optimal – one can show that the optimal strategy is a thresholding approach exactly like the one we obtained and as such, since we optimized the best threshold, the algorithm above itself is optimal (and all the equations in the analysis are approximate equalities).

3 The Prophet Inequality

We now switch to the prophet inequality setting. Here, we have an ordered set of random variables $X_1, \ldots, X_n$ with known distributions. At each time step t, we get to see the sample $x_t \sim X_t$. Then, we can either decide to *choose* x_t and obtain a value of x_t and terminate, or *pass on* x_t and continue to the next time step. The goal is to maximize the *expectation* of the value we choose. The benchmark we compare ourself with is the expected value of $\max_{i \in [n]} X_i$. I.e., if ALG denotes the value we obtain in this process, our goal is to achieve

$$\mathbb{E}\left[ALG\right] \geqslant \alpha \cdot \mathbb{E}\left[\max_{i \in [n]} X_i\right], \tag{1}$$

for the largest possible choice of $\alpha \in [0, 1]$.

The equalities of the above type in this setting are called **Prophet Inequalities** the idea being that, even with the limited information, one can still come close to the performance of a prophet who is clairvoyant.

An upper bound on α. Let us first argue that α cannot be any constant more than $1/2$. Consider the following input: X_1 is always 1 deterministically, while, $X_2 = 0$ with probability $1 - \varepsilon$ and $X_2 = 1/\varepsilon$ with the remaining ε probability for some small $\varepsilon > 0$ going to zero. Now, firstly,

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

On the other hand, there are really only two possible strategies for this instance: either always pick X_1 (which is always 1 and hence there is no uncertainty about $x_1 \sim X_1$) or always pick X_2. Either way,

$$\mathbb{E}\left[ALG\right] = 1.$$

This means that, for $\varepsilon \to 0$ in the limit,

$$\mathbb{E}\left[ALG\right] \leqslant (1/2 + o(1)) \cdot \mathbb{E}\left[\max_{i \in \{1,2\}} X_i\right],$$

hence, one cannot hope for a value of $\alpha > 1/2$ in Eq (1).

An optimal prophet inequality. We are now going to show that one can also achieve $\alpha = 1/2$ in Eq (1) which is optimal in light of the above example.

To some extent, the algorithm here is also quite similar to the one for the secretary problem. In the secretary problem, we used the a part of the arriving elements to "learn" a threshold and then picked the first arriving element better than this threshold. In the prophet setting, it is clear that we should not simply discard some of the elements as the order of $X_1, \ldots, X_n$ is not random. On the other hand, we do know the distribution of $X_1, \ldots, X_n$ and can thus use this knowledge to learn a "good" threshold.

Define the random variable $X_{max} := \max_{i \in [n]} X_i$. Our goal is to achieve a value close to $\mathbb{E}\left[X_{max}\right]$. Pick the threshold θ such that

$$\Pr\left(X_{max} \geqslant \theta\right) = \frac{1}{2},$$

i.e., let θ be the *median* of the distribution of X_{max}. For simplicity, we assume that X_{max} has no point mass on θ, i.e., $\Pr\left(X_{max} < \theta\right) = 1/2$ also (we can easily fix this in the argument, but we shall skip that for sake of a simpler exposition).

The algorithm is now as follows: For $i = 1$ to n, pass on each realization $x_i \sim X_i$ if $x_i < \theta$; otherwise, if $x_i \geqslant \theta$, choose x_i and terminate.

Lemma 3. *Let ALG denote the value obtained by the above algorithm. Then,*

$$\mathbb{E}[ALG] \geqslant \frac{1}{2} \cdot \mathbb{E}[X_{max}].$$

Proof. We first upper bound the optimal solution as follows:

$$\begin{aligned}
\mathbb{E}[X_{max}] &= \mathbb{E}[\theta + X_{max} - \theta] && \text{(by adding and subtracting } \theta \text{ trivially)}\\
&\leqslant \mathbb{E}[\theta + \max(X_{max} - \theta, 0)] && \text{(as } X_{max} - \theta \leqslant \max\{X_{max} - \theta, 0\}\text{)}\\
&= \theta + \mathbb{E}[\max(X_{max} - \theta, 0)] && \text{(by linearity of expectation and since } \theta \text{ is deterministic)}\\
&\leqslant \theta + \mathbb{E}\left[\sum_{i=1}^{n} \max(X_i - \theta, 0)\right]. && \text{(by the upper bounding the `max'-term of } X_{max} \text{ with a `sum')}
\end{aligned}$$

On the other hand, we can lower bound the value chosen by the algorithm as follows. Define F as the random variable which is θ if the algorithm chooses any element and is zero otherwise. Define E as the random variable which is $ALG - F$ is the algorithm chooses any element and is zero otherwise. Think of F as the 'fixed' part of the value of ALG which is θ and E as the 'excess' part that it achieves over θ. We have,

$$\begin{aligned}
\mathbb{E}[ALG] &= \mathbb{E}[F] + \mathbb{E}[E] && \text{(as } ALG = F + E \text{ and by linearity of expectation)}\\
&= \theta \cdot \Pr(ALG \geqslant \theta) + \mathbb{E}[E] && \text{(by the definition of } F\text{)}\\
&= \theta \cdot \Pr(X_{max} \geqslant \theta) + \mathbb{E}[E].\\
& && \text{(as whenever } X_{max} \geqslant \theta\text{, the algorithm will output a value at least as large as } \theta\text{)}
\end{aligned}$$

We now also calculate $\mathbb{E}[E]$ separately. Notice that the events that the algorithm picks i-th element for $i \in [n]$ are disjoint from each other. Thus,

$$\begin{aligned}
\mathbb{E}[E] &= \sum_{i=1}^{n} \Pr(ALG = X_i) \cdot \mathbb{E}[E \mid ALG = X_i] && \text{(by the law of total expectation)}\\
&= \sum_{i=1}^{n} \Pr(ALG = X_i) \cdot \mathbb{E}[X_i - \theta \mid ALG = X_i]\\
& && \text{(if the algorithm picks } X_i\text{, the value of } E \text{ will be } X_i - \theta\text{)}\\
&= \sum_{i=1}^{n} \Pr(X_1, \ldots, X_{i-1} < \theta \wedge X_i \geqslant \theta) \cdot \mathbb{E}[X_i - \theta \mid ALG = X_i]\\
& && \text{(the criteria for } ALG = X_i \text{ is if this is the first element with value } \geqslant \theta\text{)}\\
&= \sum_{i=1}^{n} \Pr(X_1, \ldots, X_{i-1} < \theta) \cdot \Pr(X_i \geqslant \theta) \cdot \mathbb{E}[X_i - \theta \mid ALG = X_i]\\
& && \text{(the random variables } X_1, \ldots, X_i \text{ are independent of each other)}\\
&\geqslant \sum_{i=1}^{n} \Pr(X_{max} < \theta) \cdot \Pr(X_i \geqslant \theta) \cdot \mathbb{E}[X_i - \theta \mid ALG = X_i]\\
& && \text{(the event } X_{max} < \theta \text{ is a subset of the event } X_1, \ldots, X_{i-1} < \theta\text{)}\\
&= \Pr(X_{max} < \theta) \cdot \sum_{i=1}^{n} \Pr(X_i \geqslant \theta) \cdot \mathbb{E}[X_i - \theta \mid ALG = X_i]\\
&= \Pr(X_{max} < \theta) \cdot \sum_{i=1}^{n} \Pr(X_i \geqslant \theta) \cdot \mathbb{E}[X_i - \theta \mid X_i \geqslant \theta]\\
& && \text{(we have } \mathbb{E}[X_i \mid X_i \geqslant \theta] = \mathbb{E}[X_i \mid ALG = X_i] \text{ as } X_i \text{ is independent of } X_1, \ldots, X_{i-1}\text{)}\\
&= \Pr(X_{max} < \theta) \cdot \sum_{i=1}^{n} \mathbb{E}[\max(X_i - \theta, 0)],
\end{aligned}$$

since we have,

$$\begin{aligned}\mathbb{E}\left[\max\left(X_i-\theta,0\right)\right] &= \Pr\left(X_i \geqslant \theta\right)\cdot\mathbb{E}\left[\max\left(X_i-\theta,0\right)\mid X_i\geqslant\theta\right]+\Pr\left(X_i<\theta\right)\cdot\mathbb{E}\left[\max\left(X_i-\theta,0\right)\mid X_i<\theta\right]\\ &= \Pr\left(X_i\geqslant\theta\right)\cdot\mathbb{E}\left[X_i-\theta\mid X_i\geqslant\theta\right]+0.\end{aligned}$$

Plugging in the bounds obtained for F and E in the $ALG = E + F$ equation above, we get that,

$$\begin{aligned}\mathbb{E}\left[ALG\right] &\geqslant \theta\cdot\Pr\left(X_{max}\geqslant\theta\right)+\Pr\left(X_{max}<\theta\right)\cdot\sum_{i=1}^{n}\mathbb{E}\left[\max\left(X_i-\theta,0\right)\right]\\ &= \frac{1}{2}\cdot\left(\theta+\mathbb{E}\left[\sum_{i=1}^{n}\max\left(X_i-\theta,0\right)\right]\right) && \text{(by the choice of } \theta \text{ as the median-value)}\\ &\geqslant \frac{1}{2}\cdot\mathbb{E}\left[X_{max}\right]. && \text{(by the upper bound on } \mathbb{E}\left[X_{max}\right] \text{ calculated earlier)}\end{aligned}$$

This concludes the proof. □

This concludes our study of basic online decision making problems in this course.