Pandora's box

BY C. DÜRR, MPRI 2-24-1, OCT 2021

Here is an old and fundamental problem, introduced by Martin Weitzman in 1979. We are given n boxes. Each box i has an opening cost $c_i > 0$ and a hidden value X_i, which is drawn randomly from a known distribution. The goal is to open a set of boxes S and to maximize the profit, which is defined as the maximum value among all boxes in S minus the total opening cost of all boxes in S. The idea is that you first open a few boxes and then select the best valued opened box.

$$\begin{array}{ccc} \boxed{X_1} & \dots & \boxed{X_n} \\ c_1 & & c_n \end{array}$$

An algorithm for this problem specifies an order in which the boxes will be opened, and a stopping rule, specifying, when it is done with opening boxes. The order could be chosen adaptively according to observed values, but as we will see, adaptivity does not help for this problem. Contrary to the Prophet inequality problem or the rent-or-buy problem, we will not measure the quality of an algorithm by its competitive ratio. Our approach is more direct, and we will determine the optimal algorithm for this problem, in terms of expected profit.

The first idea that comes into mind is simply to remove the opening costs from the random variables, and to search for the box with the maximum expected modified value. This works well for a single box. Indeed, if you open it, the expected profit is $\mathbb{E}[X_1] - c_1 = \mathbb{E}[X_1 - c_1]$ and if you don't open the box, your profit is just zero. This, by the way, means that we can assume without loss of generality that every box satisfies $c_i < \mathbb{E}[X_i]$, otherwise it is safe to ignore it.

But for more boxes, this approach fails. Because the profit depends on the maximum among the values of the opened boxes and on the sum of their opening costs. So these quantities have to be handled differently and cannot be combined into a single quantity. Note that if the goal were to maximize the total value of the opened boxes minus the total opening costs, then by linearity, each box would represent an independent subproblem, which can be solved trivially as we just observed.

1 Key idea: splitting

The essence of the problem lays in the following situation. We already opened some boxes, and observed a maximum value y. Should we open box i or not? It would only make sense if X_i exceeds y, because then we observe an increase in the profit of $X_i - y$, from which however we need to pay the opening cost c_i. Hence only if $X_i - y$ exceeds c_i, then we would be happy to open box i. Formally we need to compare y with $\mathbb{E}[\max\{y, X_i\}] - c_i$. So if y is small enough, we would like to open the box. If y is large enough, we better not open the box. In between we are indifferent.

In order to manipulate more easily these expressions, we split X_i into two parts, a capped value and a bonus. Formally, we use a threshold σ_i, which will be specified in a moment, and set $X_i = \kappa_i + b_i$, where the capped value κ_i is $\min\{\sigma_i, X_i\}$ and the bonus b_i is $X_i - \kappa_i$. So for a fixed σ_i, we split X_i into two new random variables. Now we define σ_i to be such that $\mathbb{E}[b_i] = c_i$.

This threshold is called the *fair cap*, or sometimes also the *reservation price*.

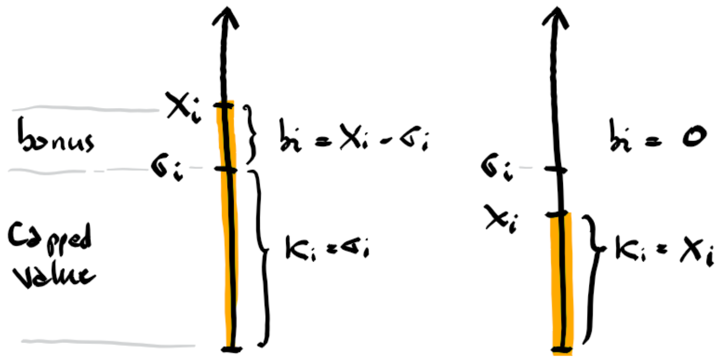

Figure 1. Splitting the value of box i

Remember, we were supposed to answer the question, whether to open box i or not. Well, we should open it if $y < \sigma_i$. But this does not tell us yet in what order we should open the boxes.

2 Upper bound

We provide now an upper bound on the expected profit of any algorithm. Understanding this upper bound will lead directly to the optimal algorithm. To be more formal, we introduce binary decision variables $I_i, A_i \in \{0,1\}$ for every box, describing succinctly the behavior of an algorithm.

- I_i=1 if algorithm opened box i. Letter I stands for inspection.
- $A_i = 1$ if algorithm selected box i. A does not stand for anything particular. The choice of this letter just creates confusion if I_i and A_i are not pronounced carefully.

So if the algorithm does not open any box, then all these variables are zero. Otherwise we have $A_i = 1$ for the maximizer of X_j over all j with $I_j = 1$.

We say that an algorithm *claims above the cap*, if whenever it opens a box ($I_i = 1$) and observes a value exceeding the cap ($b_i > 0$), then it selects this box ($A_i = 1$). In particular it means that such an algorithm cannot open two boxes with a positive bonus, since it must stop after opening the first box with positive bonus. The class of these algorithms is quite large, it does not specify anything about the order in which boxes are opened, and contains also algorithms selecting boxes with zero bonus, or even the trivial algorithm which does not open any box.

Lemma 1. *Consider an arbitrary algorithm and denote by ALG the random profit it generates. Then*

$$\mathbb{E}[\mathrm{ALG}] \leq \sum_{i=1}^{n} \mathbb{E}[A_i \kappa_i],$$

and we have equality if and only if the algorithm claims above the cap.

Proof. The following sequence of inequalities, uses linearity of expectation, the equality $\mathbb{E}[b_i] = c_i$, but most importantly the fact that I_i and b_i are independent. Indeed the algorithm's decision to open box i cannot depend on its value. This means that $\mathbb{E}I_i\mathbb{E}b_i = \mathbb{E}I_i b_i$. We have

$$\begin{aligned}
\mathbb{E}[\mathrm{ALG}] &= \sum_{i=1}^{n} \mathbb{E}[A_i X_i - I_i c_i] \\
&= \sum_{i=1}^{n} \mathbb{E}[A_i(\kappa_i + b_i) - I_i \mathbb{E}[b_i]] \\
&= \sum_{i=1}^{n} (\mathbb{E}[A_i(\kappa_i + b_i)] - \mathbb{E}I_i \mathbb{E}b_i) \\
&= \sum_{i=1}^{n} (\mathbb{E}[A_i(\kappa_i + b_i)] - \mathbb{E}I_i b_i) \\
&= \sum_{i=1}^{n} \mathbb{E}[A_i(\kappa_i + b_i) - I_i b_i] \\
&= \sum_{i=1}^{n} \mathbb{E}[A_i \kappa_i - (I_i - A_i) b_i].
\end{aligned}$$

The claimed inequality follows from $A_i \geqslant I_i$ and $b_i \geqslant 0$. Moreover we have equality only of $(I_i - A_i) b_i = 0$ for every box i, meaning that if the algorithm opens box i ($I_i = 1$) and observes a positive bonus ($b_i > 0$), then it selects this box ($A_i = 1$). In other words we have equality if and only if the algorithm claims above the cap. ☐

We learn two things from this bound. First that we need to search for an optimal algorithm among the algorithms claiming above the cap. But also that no algorithm can exceed in expectation $\mathbb{E}[\max_i \kappa_i]$.

3 The optimal algorithm

Let's try to design an algorithm selecting the box with the maximum capped value. By the previous observation, this would be the optimal algorithm. Let's step back and see what problem we have to solve now. In fact we just have a bunch of random variables $\kappa_1, \ldots, \kappa_n$, and all we know is that $\kappa_i \in [0, \sigma_i]$. Our goal is to open box i with the maximum κ_i, under the constraint that once we open a box i with $\kappa_i = \sigma_i$ (i.e. $b_i > 0$), we need to claim that box and stop. We ended up in this simple reformulation of our problem, and got rid of opening costs.

Observe that if $\sigma_i > \sigma_j$, then it is very risky to open box j before i, because we might have to claim box j, and it could be that $\kappa_i > \sigma_j$. This leads to the following optimal algorithm. For convenience sort the boxes in non-increasing order of fair caps, that is $\sigma_1 \geqslant \ldots \geqslant \sigma_n$.

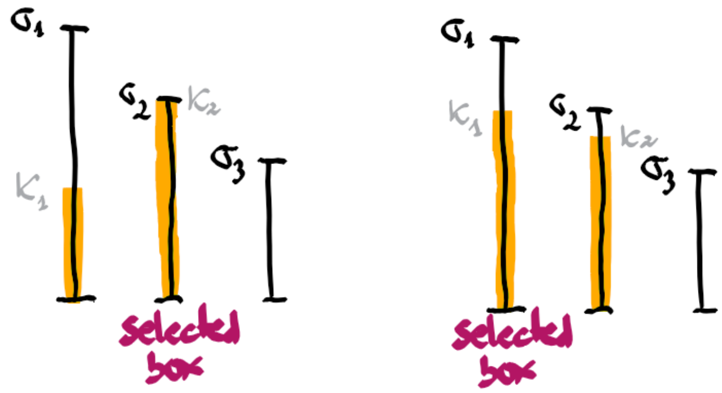

Figure 2. Examples of possible runs of the algorithm

Open the boxes in this order. Once we opened boxes $1, \ldots, j$ with a maximum observed capped value κ_i such that $\kappa_i \geqslant \sigma_{j+1}$, we stop and claim box i. This happens in particular if $\kappa_i = \sigma_i$, i.e. when we are forced to claim box i.

Observe that the algorithm is defined in such a way that it selects a box with the maximum capped value, even among the boxes it did not open. Hence this algorithm is optimal.

Sanity check: imagine a situation with two boxes of same fair cap $\sigma_1 = \sigma_2$, meaning that there are two possible valid orders of the boxes. In case $\kappa_1 < \kappa_2 = \sigma_2$, for the order $(1, 2)$ the algorithm will open both boxes, while for the other order, the algorithm will open only a single box. But this is ok, as in the new formulation of the problem, we don't care about opening costs. Reaching a box with maximum capped value is enough.

4 Hidden details

So far we made no assumption on the distribution from which the random variables X_i are drawn. In order to ensure that the fair cap σ_i exists, we need that the distribution has no probability mass at any point. Simply because we want the function $y \mapsto \mathbb{E}[X_i - \min\{y, X_i\}]$ to be continuous. Also if it exists, in principle there might be a whole interval of valid fair caps. In that case any choice would be good.

Also in the definition of the algorithm we used the fact that if the capped value satisfies $\kappa_i = \sigma_i$ then the bonus is positive, i.e. $b_i > 0$. We have this property with probabilty 1, if the distributions of the values X_i have no probability mass at point σ_i.

However the algorithm presented in this note can be adapted to discrete distributions and works similarly.