Online Algorithms with Costly Predictions

Marina Drygala[a], **Sai Ganesh Nagarajan**[a], **Ola Svensson**[a]
[a]EPFL, Switzerland

Abstract. In recent years there has been a significant research effort on incorporating predictions into online algorithms. However, work in this area often makes the underlying assumption that predictions come for free (e.g., without any computational or monetary costs). In this work, we consider a cost associated with making predictions. We show that interesting algorithmic subtleties arise for even the most basic online problems, such as ski rental and its generalization, the Bahncard problem. In particular, we show that with costly predictions, care needs to be taken in (i) asking for the prediction at the right time, (ii) deciding if it is worth asking for the prediction, and (iii) how many predictions we ask for, in settings where it is natural to consider making multiple predictions. Specifically, (i) in the basic ski-rental setting, we compute the optimal delay before asking the predictor, (ii) in the same setting, given apriori information about the true number of ski-days through its mean and variance, we provide a simple algorithm that is near-optimal, under some natural parameter settings, in deciding if it is worth asking for the predictor and (iii) in the setting of the Bahncard problem, we provide a $(1+\varepsilon)$-approximation algorithm and quantify lower bounds on the number of queries required to do so. In addition, we show that solving the problem optimally would require almost complete information of the instance.

Summary of Contributions

Many classic online problems are studied under the umbrella of learning-augmented algorithms. In this setting, we have access to a ML predictor appropriate for the problem. Algorithms that use these ML predictions must ensure that, when the predictor is correct, the performance should match with that of the offline optimum and gracefully degrade to the best online algorithm when the ML predictions are unreliable. Existing work assumes that predictions from the ML oracle are free of cost. However, in practice, it is natural to consider scenarios where obtaining ML predictions may incur costs that are either computational or monetary due to necessary data collection processes. This potentially restricts the amount of calls one could make to the ML oracle to solve the problem at hand and if perhaps the cost is too high it may not be helpful to even ask for advice. This perspective introduces new aspects that differ from standard learning augmented algorithms. That is, without a costly predictor, it is *always* in our best interest to ask the predictor at the beginning of an online prediction interval ($t = 0$). However, this need not be the case when the prediction carries a cost. To this end, we conceptualize three fundamental questions when predictions carry a cost. (i) When do we ask the predictor? (ii) Given apriori information about the problem, such as certain useful statistics, should we make use of the predictor at all? (iii) How often should we ask the predictor to collect the required information to solve a problem either optimally or approximately? We remark, that question (iii) has been studied very recently in the setting of paging by.[1]

In this work,[2] we study the phenomenon of costly predictions for online problems through the classic ski-rental problem for questions (i) and (ii). For question (iii), we require a setting where we would have repeated calls to the predictor and we therefore consider the bahncard problem, which is a well-studied and natural generalization of the ski-rental problem to a repeated horizon setting.[3]

Ski-Rental and the Bahncard Problem

In an instance of the ski-rental problem we will ski for an unknown number of days t. On the beginning of each day, an irrevocable decision of whether to rent or buy skis must be made. Skis can be rented each day at cost 1 or bought for use during the remainder of the season at cost b. The offline optimum is easily computed once the duration of our ski season is revealed. It is well known that there is a simple deterministic buying strategy that achieves a competitive ratio of 2.[4] Additionally,[5] provide a randomized algorithm that achieves a competitive ratio of $\frac{e}{e-1}$. Both of the above algorithms are tight. The bahncard problem can be viewed as a generalization of the ski-rental problem to a repeated horizon setting. Specifically, each day we take train trips of a given cost (potentially 0). If we do not have a valid bahncard at any given point, we have to make an online irrevocable decision of whether or not to purchase one at cost B. If we do not purchase one, we must pay full price for that day's tickets. If we have a valid bahncard on a given day, then we get a discount of $\beta \in [0, 1]$ for our ticket (i.e, discounted cost is $\beta \cdot$ original cost). Each bahncard is valid for T days. An algorithm for the bahncard problem should output the sequence of purchasing times for the bahncards. It was shown by[3] that there is a simple deterministic buying strategy that achieves a competitive ratio of $2 - \beta$. Finally,[6] provide a randomized algorithm that achieves a competitive ratio of $\frac{e}{e-1+\beta}$. Note that when $\beta = 0$ and $B = b$ and $T \to \infty$, this reduces to the ski-rental problem.

Ski-rental with Costs and Prior Information: For the first question, we provide a simple algorithm that waits for an optimum amount of time (roughly $\sqrt{cb}$ days) to ask for the prediction and following it's advice thereafter. We show that this algorithm is optimal by minimizing the competitive ratio whilst considering the cost of asking for a prediction. To address the second question, we propose a simple deterministic algorithm which is near-optimal under some natural parameter settings in deciding whether to ask the predictor, when only the mean and the variance is known. We show that there is a threshold function $f^*(\mu, \sigma, b)$, that essentially computes the "value" of the prediction. For instance, if $\mu = b$, for a fixed c (say $\sqrt{b}$), we have that as σ varies and crosses c, the uncertainty on the worst-case distribution becomes sub-optimal for any algorithm and in which case it would be better to ask the predictor. We can also make our algorithms robust to prediction errors by using standard techniques from.[7] .

Bahncard problem with Few Predictions: Our main technical contributions are on the bahncard problem, a generalization of the ski-rental problem to a repeated horizon setting. When we associate a cost to each prediction, it is natural to ask how many predictions are required to gather enough information to output a buying schedule that is close to optimal. To this end, we characterize the query complexity of solving this problem both optimally and to a factor of $1+\varepsilon$. We provide upper and lower bounds on the number of queries required to achieve a $(1 + \varepsilon)$-approximation algorithm. The upper bound given by our algorithm is nearly tight. Our approach in this part is based on several novel ideas and is more technically advanced compared to the simple and clean algorithms that we analyze for the ski-rental problem. In particular, we heavily exploit structural properties of the bahncard problem to compute intervals of possibly high cost, that we need not obtain information on in order to compute a good solution. Finally, we describe how to modify our algorithm to accommodate prediction errors. This modification requires new ideas on how to partition the timeline to appropriately to charge costs and prediction errors.

References

1 S. Im, R. Kumar, A. Petety, *et al.*, "Parsimonious learning-augmented caching," *arXiv preprint arXiv:2202.04262* (2022).

2 M. Drygala, S. G. Nagarajan, and O. Svensson, "Online algorithms with costly predictions," in *International Conference on Artificial Intelligence and Statistics*, 8078–8101, PMLR (2023).

3 R. Fleischer, "On the bahncard problem," *Theoretical Computer Science* **268**(1), 161–174 (2001).

4 A. R. Karlin, M. S. Manasse, L. Rudolph, *et al.*, "Competitive snoopy caching," *Algorithmica* **3**(1), 79–119 (1988).

5 A. R. Karlin, M. S. Manasse, L. A. McGeoch, *et al.*, "Competitive randomized algorithms for non-uniform problems," in *Proceedings of the first annual ACM-SIAM symposium on Discrete algorithms*, 301–309 (1990).

6 A. R. Karlin, C. Kenyon, and D. Randall, "Dynamic tcp acknowledgement and other stories about e/(e-1)," in *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, 502–509 (2001).

7 M. Purohit, Z. Svitkina, and R. Kumar, "Improving online algorithms via ml predictions," *Advances in Neural Information Processing Systems* **31**, 9661–9670 (2018).