SOFSEM PhD School: Algorithms with Predictions

Exercise Sheet

Adam Polak, Christian Coester February 9, 2026

Exercise 1. Randomized ski rental. Design a randomized learning-augmented algorithm for the ski rental problem that achieves consistency and robustness unachievable for any deterministic learning-augmented algorithm. Recall that, for every $w \geq 2$, any w-robust deterministic algorithm can be at best $\frac{w}{w-1}$-consistent.

Hint: Consider the following algorithm: if the predicted season length y is less than the buying cost B, wait B days and buy if the season is still not over; whereas if $y \geq B$, flip a fair coin, and wait either $B/3$ or $2B/3$ days before buying. What is the consistency and robustness of this algorithm?

Exercise 2. Existence of good weights. Consider the following routing problem: At each time $t = 1, 2, \ldots, T$, a set P_t of paths in an underlying directed graph $G = (V, E)$ is revealed, and we have to send one unit of flow (split arbitrarily) along the paths in P_t. The *congestion* of an edge is the total amount of flow sent through it across all time steps. The goal is to minimize the congestion of the most congested edge.

Show that for any instance $P_1, \ldots, P_T$ and $\epsilon > 0$, there exist edge weights w_a for $a \in E$ such that the following strategy is within a factor $1 + \epsilon$ of optimal: For each path $p \in P_t$, allocate a flow of

$$\frac{\prod_{a \in p} w_a}{\sum_{q \in P_t} \prod_{a \in q} w_a}$$

along path p at time t.

Exercise 3. Combining algorithms. In the online bipartite metric matching problem, there is an underlying metric space (M, d) and a set $S = \{s_1, \ldots, s_n\} \subseteq M$ of servers is known in advance. A sequence of requests $r_1, r_2, \ldots, r_n \in M$ arrives online, one by one. Upon arrival of request r_t, the algorithm must immediately and irrevocably match it to an unmatched server $s \in S$, incurring cost $d(r_t, s)$. The goal is to minimize the total cost.

Show that for any two online algorithms A and B for this problem, there exists an online algorithm C whose cost is at most 9 times the cost of the better of the two algorithms, on any instance, i.e.,

$$\mathrm{cost}_C \leq 9 \cdot \min\{\mathrm{cost}_A, \mathrm{cost}_B\}.$$

Hint. Using the 9-competitive cow-path strategy, try to *simulate* following one of the algorithms at a time, but note that truly switching configurations is impossible due to the irrevocability of decisions. Let $A_t, B_t, C_t \subseteq S$ be the sets of matched servers by A, B, C after the t-th request. Maintain a bijection $b_t \colon C_t \to F_t$ where $F \in \{A, B\}$ is the algorithm we are currently trying to follow, and use

$$\Phi_t := \sum_{s \in C_t} d(s, b_t(s))$$

as a potential function. Show for each step following F, one can achieve $\Delta\mathrm{cost}_C + \Delta\Phi \leq \Delta\mathrm{cost}_F$, and when F switches between A and B, one can update the bijection such that $\Delta\Phi \leq \mathrm{cost}_A + \mathrm{cost}_B$.

Exercise 4. Sorting. A *finger tree* is a binary search tree with a pointer ("finger") to the last inserted item. When the finger points to u, it supports the operation of inserting a new value v in amortized time $O(\log d(u,v))$, where $d(u,v)$ is the number of items in the tree whose value lies in the range between u and v (inclusive); after the insertion, the finger points to v.

We want use a finger tree to sort a list of items $a_1, \ldots, a_n$. The algorithm's input is augmented with predictions $\hat{p}(i)$ for the position $p(i)$ of a_i in the sorted list. Let $\eta_i := |p(i) - \hat{p}(i)|$ be the prediction error of item a_i. By bucket sorting in time $O(n)$ according to $\hat{p}$, we may assume without loss of generality that $\hat{p}(1) \leq \hat{p}(2) \leq \cdots \leq \hat{p}(n)$.

Show that the following procedure can be used to sort the items in time $O\left(\sum_{i=1}^{n} \log(\eta_i + 2)\right)$.

```
T ← empty finger tree
for i = 1, ..., n do                ▷ Assume p̂(1) ≤ p̂(2) ≤ ··· ≤ p̂(n) by bucket-sort
    Insert a_i into T
```

Bonus question: Can you describe a sorting algorithm with the same running that works *in-place* (i.e., *without* requiring extra space $\Omega(n)$ on top of the input array)?

Exercise 5. Learning matching duals. Let $G = (V, E)$ be a fixed bipartite graph, let $U \in Z_+$ be an upper bound of edge weights, and let $\mathcal{D}$ be a distribution over vectors of integer U-bounded edge weights $[U]^E$.

1. Let $y \in [U]^V$ be a fixed dual solution. Show that, for every $\epsilon, p > 0$, for a large enough number of samples $w_1, w_2, \ldots, w_k \sim \mathcal{D}$, with probability at least, $1 - p$, the *empirical* prediction error
$$\frac{1}{k} \sum_{i \in [k]} ||y - y^*(w_i)||_1$$
is within $(1 + \epsilon)$ factor from the true prediction error
$$\mathbb{E}_{w \sim \mathcal{D}} ||y - y^*(w)||_1.$$
Hint: use the Hoeffding bound.

2. Show that, for every $\epsilon, p > 0$, for a large enough number of samples $w_1, w_2, \ldots, w_k \sim \mathcal{D}$, with probability at least, $1 - p$, for every dual solution $y \in [U]^V$, the empirical prediction error is within $(1 + \epsilon)$ factor from the true prediction error. Hint: repeat the previous argument for smaller p and use a union bound over all dual solutions. (Side note: here we use the fact that the number of dual solutions is finite, which follows from the fact that we work with integer weights; if we worked with real weights we would need to argue about the VC-dimension.)

3. Conclude that, a dual minimizing the empirical prediction error is, with good probability over the choice of samples, an approximate minimizer of the true prediction error.

4. Show that the element-wise median of $y^*(w_1), y^*(w_2), \ldots, y^*(w_k)$ minimizes the empirical prediction error.