Learning Predictions for Algorithms with Predictions
Misha Khodak, Nina Balcan, Ameet Talwalkar, Sergei Vassilvitskii
khodak@cmu.edu

Algorithms with predictions
take advantage of a prediction \( w \) to improve the cost \( C_x(w) \) of running on an instance \( x \)
generic guarantee: \( C_x(w) \) is bounded by a function \( U_x(w) \), which is
1. small if prediction \( w \) is good (consistency)
2. as good as the worst-case (robustness)

A new framework for learning predictions

Step 1: derive a “nice” upper bound \( U_x(w) \) of \( C_x \)
- \( U_x \) should be a surrogate loss for \( C_x \)
- convex, Lipschitz, etc.

Step 2: apply online learning
- low regret: \( \sum_t U_x(w_t) - \min_w \sum_t u_x(w) = o(T) \)
- sample complexity: \( E_p U_x(\theta) \leq \min_w E_p U_x(w) + \varepsilon \)
- instance-dependent prediction: \( w \leftarrow (\alpha, f_{\alpha}) \)
- problem-specific learning algorithms

Our framework

Problem: for a bipartite graph with \( m \) edges and \( n \) vertices, find the perfect matching with least weight according to edge-weights \( x \in \mathbb{Z}^n \)

Example: Bipartite matching

Algorithm [1]: Hungarian method initialized at integer duals \( w \in \mathbb{Z}^n \) has runtime \( O(m \sqrt{n} \| w^* - y^*(x) \|_1) \), where \( y^*(x) \in \mathbb{Z}^n \) is the dual vector of the optimum

Step 1: rounding \( w \in \mathbb{R}^n \) to the integers before running Hungarian
- preserves distance to \( y^*(c) \) up to constants
- makes the problem of learning \( w \) convex

Step 2: apply online gradient descent to \( U_x(w) = \| w - y^*(x) \|_1 \)
- \( \tilde{O}(n^2/\varepsilon^2) \) samples needed to PAC-learn \( w \)
- \( O(n \sqrt{T}) \) cumulative regret

\( O(n) \) improvement over [1]

References:
[1] Dinitz, Im, Lavastida, Moseley, Vassilvitskii, NeurIPS 2021
[3] Indyk, Mallmann-Trenn, Mitrovic, Rubinfeld, AISTATS 2022

More applications
Better bounds for shortest path and \( b \)-matching
We extend our matching guarantees to obtain up to \( O(n^{-2}) \) improvement in sample complexity over [2]

First learning guarantees for online page migration
Goal: predict a distribution over a finite metric space \( K \) to satisfy a sequence of \( n \) requests
Step 1: make existing guarantee [3] convex at cost \( O(\log n) \)
Step 2: apply exponentiated gradient descent
- regret: \( O(n \sqrt{T} \log |K|) \)
- sample complexity: \( O(n^2 \log |K|) \)

Tuning robustness-consistency tradeoffs
Robustness-consistency can be traded off parametrically: \( C_x(w, \lambda) \leq \min_{f} (f) U_x(w), g(\lambda)) \)
for \( f \) increasing, \( g \) decreasing, and \( \lambda \in [0, 1] \).
We show how to learn the best \( \lambda \) using data, sometimes simultaneously with learning the prediction.

Learning predictions for job scheduling
See paper (arxiv.org/abs/2202.09312) for learning predictions
- that improve the fractional makespan in online scheduling
- of optimal job permutations for non-clairvoyant scheduling

References:
[1] Dinitz, Im, Lavastida, Moseley, Vassilvitskii, NeurIPS 2021
[3] Indyk, Mallmann-Trenn, Mitrovic, Rubinfeld, AISTATS 2022