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

Our framework

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

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

Example: Bipartite matching

Step 1: rounding \( w \in \mathbb{R}^m \) 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

\( \tilde{O}(n) \) improvement over [1]

First learning guarantees for online page migration

Step 1: make existing guarantee [3] convex at cost \( O(\log n) \) requests
Step 2: apply exponentiated gradient descent
- regret: \( O(n\sqrt{T}\log|K|) \)
- sample complexity: \( O\left(\frac{n^2}{\varepsilon^2}\log|K|\right) \)

References:
[1] Dinitz, Im, Lavastida, Moseley, Vassilvitskii. NeurIPS 2021
[2] Chen, Silwal, Vakilian, Zhang. ICML 2022.
[3] Indyk, Mallmann-Trenn, Mitrovic, Rubinfeld, AISTATS 2022

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]

Tuning robustness-consistency tradeoffs

Robustness-consistency can be traded off parametrically: \( C_g(w, \lambda) \leq \min\{f(\lambda)^2 U_x(w), g(\lambda)\} \)
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