Improving online algorithms with ML predictions

Ravi Kumar, Manish Purohit, Zoya Svitkina

NeurIPS'18

Online algorithms

Full input not revealed upfront, but at some later stage, e.g.:

- **Matching:** nodes of a graph arrive over time
  - Must irrevocably decide whether to match a node when it arrives
- **Caching:** memory access requests arrive over time
  - Must decide what to keep in cache
- **Scheduling:** job lengths not revealed until they terminate
  - Must decide which jobs to schedule when

Competitive ratio (CR)

Standard measure of online algorithm's performance:

$$\mathrm{CR} = \frac{\mathrm{ALG}}{\mathrm{OPT}}$$

Offline optimal solution that knows the entire input

E.g., in matching:

$$\mathrm{CR} = \frac{\text{weight of algorithm's matching}}{\text{maximum weight matching}}$$

Online algorithms

Full input not revealed upfront, but at some later stage

What if algorithm receives some **predictions** about input?

- **Online advertising**

  e.g., Mahdian et al. [EC'07]; Devanur, Hayes [EC'09]; Muñoz Medina, Vassilvitskii [NeurIPS'17]
- **Caching**

  e.g., Lykouris, Vassilvitskii [ICML'18]
- **Data structures**

  e.g., Mitzenmacher [NeurIPS'18]
- This paper
- …

Outline

1. **Ski rental**
2. Job scheduling

Example: Ski rental problem

Problem: Skier will ski for unknown number of days

- Can either **rent each day** for \$1/day or **buy** for \$$b$
- E.g., if ski for 5 days and then buy, total price is $5 + b$

If ski x days, **optimal clairvoyant** strategy pays $\mathrm{OPT} = \min\{x, b\}$

Breakeven strategy: Rent for $b - 1$ days, then buy

- $\mathrm{CR} = \frac{\mathrm{ALG}}{\mathrm{OPT}} = \frac{x\mathbf{1}_{\{x<b\}} + (b-1+b)\mathbf{1}_{\{x\geq b\}}}{\min\{x,b\}} < 2$ (best deterministic)
- Randomized alg. $\mathrm{CR} = \frac{e}{e-1}$ [Karlin et al., Algorithmica '94]

Example: Ski rental problem

Prediction y of number of skiing days, error $\eta = |x - y|$

Baseline: Buy at beginning if $y > b$, else rent all days

Theorem: $\mathrm{ALG} \leq \mathrm{OPT} + \eta$

If y small but $x \gg b$, CR can be unbounded

Outline

1. Ski rental
   i. **Deterministic algorithm**
   ii. Randomized algorithm
2. Job scheduling

Example: Ski rental problem

Prediction y of number of skiing days, error $\eta = |x - y|$

Algorithm (with parameter $\lambda \in [0,1]$):

If $y \geq b$, buy on start of day $\lceil \lambda b \rceil$; else buy on start of day $\lceil \frac{b}{\lambda} \rceil$

- If **really trust** predictions: set $\lambda = 0$

  Equivalent to blindly following predictions
- If **don't trust** predictions: set $\lambda = 1$

  Equivalent to running the worst-case algorithm

Example: Ski rental problem

Prediction y of number of skiing days, error $\eta = |x - y|$

Algorithm (with parameter $\lambda \in [0,1]$):

If $y \geq b$, buy on start of day $\lceil \lambda b \rceil$; else buy on start of day $\left\lceil \frac{b}{\lambda} \right\rceil$

Theorem: Algorithm has CR $\leq \min\left\{\frac{1+\lambda}{\lambda}, 1 + \lambda + \frac{\eta}{(1-\lambda)\mathrm{OPT}}\right\}$

- If predictor is perfect ($\eta = 0$), **CR is small** ($\leq 1 + \lambda$)
- No matter how big η is, setting $\lambda = 1$ **recovers baseline** CR $= 2$

Example: Ski rental problem

Theorem: Algorithm has CR $\leq \min\left\{\frac{1+\lambda}{\lambda}, 1+\lambda+\frac{\eta}{(1-\lambda)\mathrm{OPT}}\right\}$

Proof sketch: If $y \geq b$, buys on start of day $\lceil \lambda b \rceil$

$$\frac{\mathrm{ALG}}{\mathrm{OPT}} = \begin{cases} \frac{x}{x} & \text{if } x < \lceil \lambda b \rceil \\ \frac{\lceil \lambda b \rceil - 1 + b}{x} & \text{if } \lceil \lambda b \rceil \leq x \leq b \\ \frac{\lceil \lambda b \rceil - 1 + b}{b} & \text{if } x \geq b \end{cases}$$

Worst when $x = \lceil \lambda b \rceil$ and CR = $\frac{b+\lceil \lambda b \rceil - 1}{\lceil \lambda b \rceil} \leq \frac{1+\lambda}{\lambda}$; similarly for $y < b$

Design principals

Consistency:

- Predictions are perfect ⇒ recover offline optimal
- Algorithm is α-consistent if CR → α as error $\eta \to 0$

Robustness:

- Predictions are terrible ⇒ no worse than worst-case
- Algorithm is β-consistent if CR ≤ β for all η

E.g., ski rental: CR $\leq \min\left\{\frac{1+\lambda}{\lambda}, 1 + \lambda + \frac{\eta}{(1-\lambda)\mathrm{OPT}}\right\}$

$(1+\lambda)$-consistent, $\left(\frac{1+\lambda}{\lambda}\right)$-robust

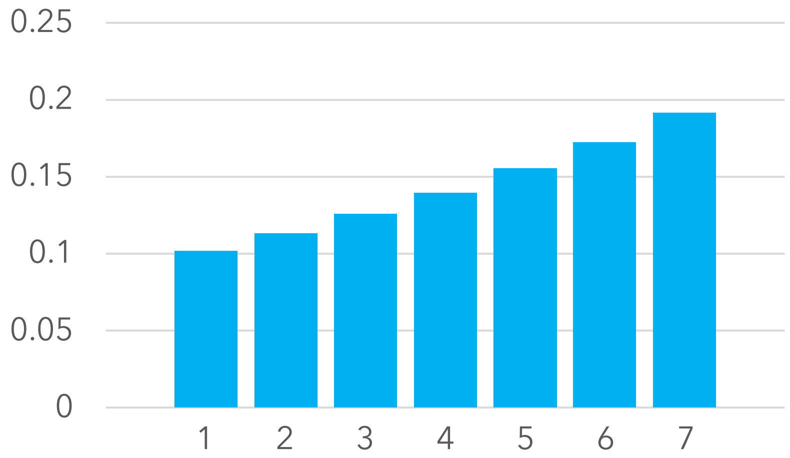
Bounds are tight [Gollapudi, Panigrahi, ICML'19; Angelopoulos et al., ITCS'20]

Lykouris, Vassilvitskii, ICML'18

Outline

1. Ski rental
   i. Deterministic algorithm
   ii. **Randomized algorithm**
2. Job scheduling

Randomized algorithm



q_i when $b = 10, \lambda = 0.7, y \geq b$

if $y \geq b$:

Let $k \leftarrow \lfloor \lambda b \rfloor$

For $i \in [k]$, define $q_i \leftarrow \left(\frac{b-1}{b}\right)^{k-i} \frac{1}{b\left(1-\left(1-{}^{1}/_{b}\right)^{k}\right)}$

Buy on day $j \in [k]$ sampled from distribution defined by $q_1, \ldots, q_k$

else

Let $\ell \leftarrow \left\lfloor \frac{b}{\lambda} \right\rfloor$

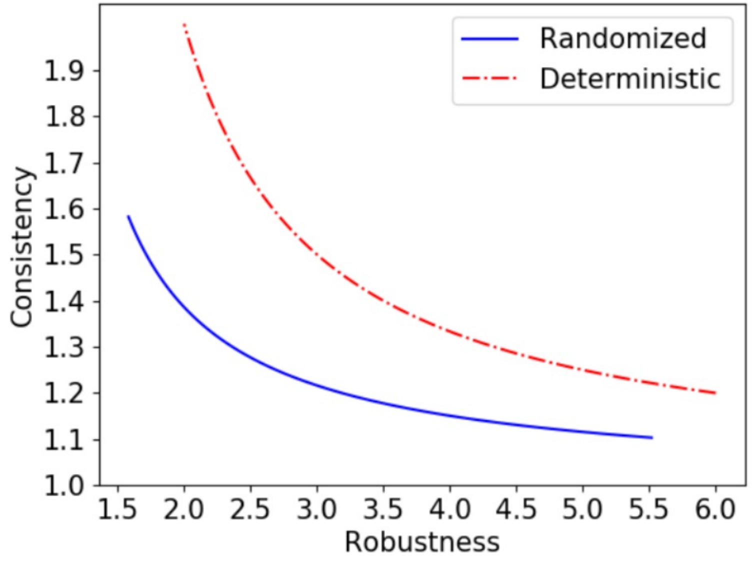
For $i \in [k]$, define $q_i \leftarrow \left(\frac{b-1}{b}\right)^{\ell-i} \frac{1}{b\left(1-\left(1-{}^{1}/_{b}\right)^{\ell}\right)}$

Buy on day $j \in [\ell]$ sampled from distribution defined by $q_1, \ldots, q_\ell$

Randomized algorithm

Theorem: $\mathrm{CR} \leq \min\left\{\frac{1}{1-\exp\left(-\left(\lambda-1/b\right)\right)}, \frac{\lambda}{1-\exp(-\lambda)}\left(1+\frac{\eta}{\mathrm{OPT}}\right)\right\}$

- $\left(\frac{\lambda}{1-\exp(-\lambda)}\right)$-consistent, $\left(\frac{1}{1-\exp\left(-\left(\lambda-1/b\right)\right)}\right)$-robust
- Bounds are **tight** [Wei, Zhang, NeurIPS'20]



Randomized algorithm

Theorem: CR $\leq \min\left\{\frac{1}{1-\exp\left(-\left(\lambda-1/b\right)\right)}, \frac{\lambda}{1-\exp(-\lambda)}\left(1+\frac{\eta}{\mathrm{OPT}}\right)\right\}$

Proof sketch:

- Split into cases depending on if $y \geq b$, $x \geq \lfloor \lambda b \rfloor$, and $x \geq \left\lfloor \frac{b}{\lambda} \right\rfloor$
- Show thm holds in each case using careful algebraic manipulations

Outline

1. Ski rental
2. **Job scheduling**

Job scheduling

Task: schedule n jobs on a single machine

Job j has **unknown** processing time x_j

Goal: minimize **sum of completion times** of the jobs

i.e., if job j completes at time c_j, goal is to minimize $\sum c_j$

Can switch between jobs

Job scheduling

Optimal solution if processing times x_j's are known:

schedule jobs in increasing order of x_j

- If $x_1 \le \cdots \le x_n$,

$$\mathrm{OPT} = \sum_{i=1}^{n} \sum_{j=1}^{i} x_j$$

Algorithm with a competitive ratio of 2: **round robin**

- Schedule 1 unit of time per remaining job, round-robin

Round-robin over k jobs $\equiv$ run jobs simultaneously at rate of $\frac{1}{k}$

Algorithms-with-predictions approach

- Predictions $y_1, \ldots, y_n$ of $x_1, \ldots, x_n$ with $\eta = \sum_{i=1}^{n} |y_i - x_i|$
- If **really trust** predictions: schedule in increasing order of y_i
  - "Shortest predicted job first (SPJF)"
- If **don't trust** predictions: round-robin (RR)

Algorithm: Preferential round-robin (with parameter $\lambda \in (0,1)$)

Run SPJF and RR **simultaneously**

- SPJF at a rate λ
- RR at a rate $1 - \lambda$

Preferential round-robin

Algorithm: Preferential round-robin (with parameter $\lambda \in (0,1)$)

Run SPJF and RR **simultaneously**

- SPJF at a rate λ
- RR at a rate $1-\lambda$

Theorem:

$$\mathrm{CR} \leq \min\left\{\frac{1}{\lambda}\underbrace{\left(1+\frac{2\eta}{n}\right)}_{\text{CR of SPJF}}, \frac{1}{1-\lambda}\cdot \underbrace{2}_{\text{CR of RR}}\right\}$$

CR of SPJF

CR of RR

Overview

Studies how to incorporate **predictions** into online algorithms

- Ski rental problem
- Job scheduling

Provable guarantees on algorithm's **competitive ratio** $\frac{\mathrm{ALG}}{\mathrm{OPT}}$

Design principals [this paper; Lykouris, Vassilvitskii, ICML'18]:

- **Consistency:** Predictions are perfect ⇒ recover offline optimal
- **Robustness:** Predictions are terrible ⇒ no worse than worst-case