LEARNING-AUGMENTED ALGORITHMS

Adam Polak

Warsaw, November 4th, 2021

Most of my work is on **fine-grained complexity**…

…but so is Karol's

Let's talk about something different

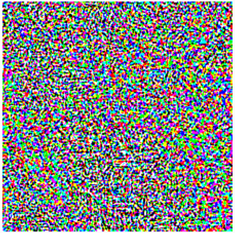
Classical algorithms

- Worst-case guarantees
- Overly pessimistic on easy instances

Machine learning

- Powerful most of the time
- No guarantees, can go crazy

$+ .007 \times$



$=$

"panda"
57.7% confidence

"nematode"
8.2% confidence

"gibbon"
99.3 % confidence

Source: arxiv.org/abs/1412.6572

Classical algorithms

- Worst-case guarantees
- Overly pessimistic on easy instances

Machine learning

- Powerful most of the time
- No guarantees, can go crazy

Best of both worlds: learning-augmented algorithms

Input + black-box predictions (e.g., coming from ML model)

MOTIVATION

Classical algorithms

- Worst-case guarantees
- Overly pessimistic on easy instances

Machine learning

- Powerful most of the time
- No guarantees, can go crazy

Best of both worlds: learning-augmented algorithms

Input + black-box predictions (e.g., coming from ML model)

- **Consistency:** close-to-optimal performance when predictions accurate

Classical algorithms

- Worst-case guarantees
- Overly pessimistic on easy instances

Machine learning

- Powerful most of the time
- No guarantees, can go crazy

Best of both worlds: learning-augmented algorithms

Input + black-box predictions (e.g., coming from ML model)

- **Consistency:** close-to-optimal performance when predictions accurate
- **Robustness:** worst-case guarantees, even when predictions adversarial

MOTIVATION

Classical algorithms

- Worst-case guarantees
- Overly pessimistic on easy instances

Machine learning

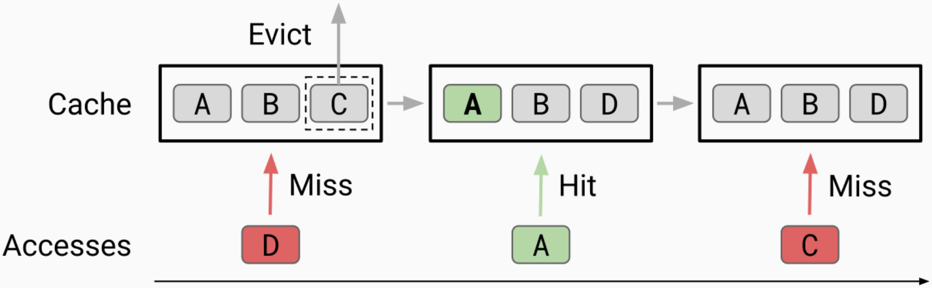
- Powerful most of the time
- No guarantees, can go crazy

Best of both worlds: learning-augmented algorithms

Input + black-box predictions (e.g., coming from ML model)

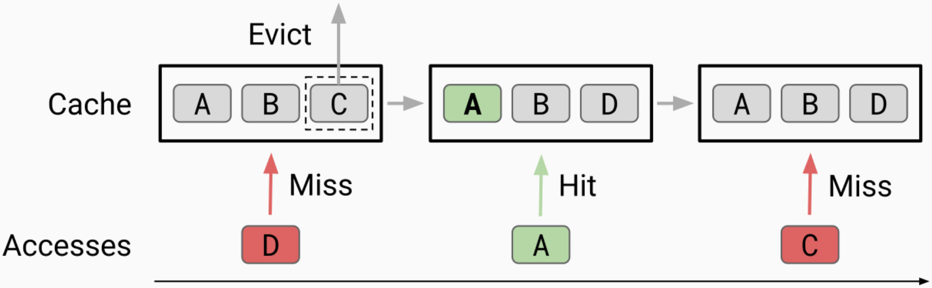
- **Consistency:** close-to-optimal performance when predictions accurate
- **Robustness:** worst-case guarantees, even when predictions adversarial
- **Smoothness:** performance degrades slowly in the prediction error

CACHING PROBLEM



Cache size: $k = 3$

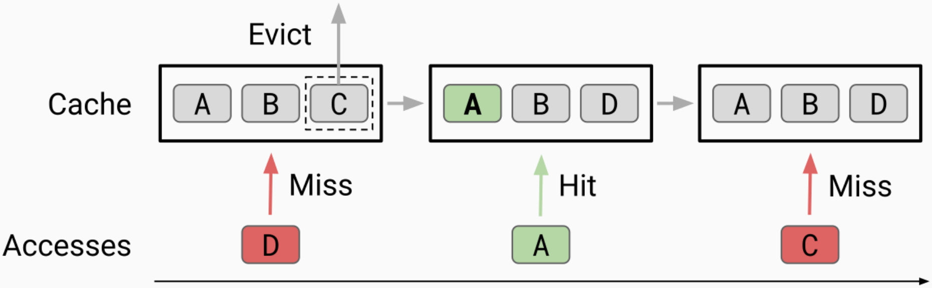
Source: arxiv.org/abs/2006.16239



Cache size: $k = 3$

Source: arxiv.org/abs/2006.16239

Offline: greedy is optimal, evict item with **largest** next arrival time (Belady, 1966)



Cache size: $k = 3$

Source: arxiv.org/abs/2006.16239

Offline: greedy is optimal, evict item with **largest next arrival time** (Belady, 1966)

Online: $O(\log k)$-competitive randomized Marker algorithm (Fiat et al., 1991)

$\text{ALG} \leqslant O(\log(k) \cdot \text{OPT})$

Theorem: (Lykouris, Vassilvitskii, ICML 2018)

Suppose each access request comes with predicted next arrival time

Prediction error $\eta = \sum_i |t_{\text{predicted}}(i) - t_{\text{real}}(i)|$

There is an $O(\min\{\sqrt{\eta/\text{OPT}}, \log k\})$-competitive algorithm

Theorem: (Lykouris, Vassilvitskii, ICML 2018)

Suppose each access request comes with predicted next arrival time

Prediction error $\eta = \sum_i |t_{\text{predicted}}(i) - t_{\text{real}}(i)|$

There is an $O(\min\{\sqrt{\eta/\mathsf{OPT}}, \log k\})$-competitive algorithm

Follow-up improvements:

- $O(\min\{\log(\eta/\mathsf{OPT}), \log k\})$ (Rohatgi, SODA 2020)
- $O(\min\{\frac{\log k}{k} \cdot \eta/\mathsf{OPT}, \log k\})$ (Rohatgi, SODA 2020)
- $O(\min\{\frac{1}{k} \cdot \eta/\mathsf{OPT}, \log k\})$ (Wei, APPROX 2020)

SOME OTHER LEARNING-AUGMENTED ONLINE ALGORITHMS

Caching (Lykouris, Vassilvitskii, ICML 2018)

Predict next arrival time

Ski rental (Purohit, Svitkina, Kumar, NeurIPS 2018)

Predict #days we will ski

Non-clairvoyant scheduling: (Purohit, Svitkina, Kumar, NeurIPS 2018)

Predict processing times

Restricted assignment (Lattanzi et al., SODA 2020)

Predict machine weights

Weighted caching (Jiang, Panigrahi, Sun, ICALP 2020)

Predict all requests until next arrival

Issue: setups tailored to specific problems

LOWER BOUND FOR WEIGHTED CACHING

Theorem: (Antoniadis, Coester, Eliáš, **P.**, Simon, ICML 2020)
(Jiang, Panigrahi, Sun, ICALP 2020)

Even with perfect predictions of next arrival times,
no better-than-classical $o(\log k)$-competitive algorithm for weighted caching

Wide class of online problems

Includes, e.g., caching, weighted caching, k-server, convex body chasing

Theorem: (Antoniadis, Coester, Eliáš, **P.**, Simon, ICML 2020)

Optimal classical competitive ratio: α (predictionless)

Suppose, at time t, given $p_t :=$ prediction of optimal algorithm's state o_t

Prediction error $\eta = \sum_t \text{dist}(p_t, o_t)$

There is an $O(\min\{\eta/\text{OPT}, \alpha\})$-competitive algorithm

Wide class of online problems

Includes, e.g., caching, weighted caching, k-server, convex body chasing

Theorem: (Antoniadis, Coester, Eliáš, **P.**, Simon, ICML 2020)

Optimal classical competitive ratio: α (predictionless)

Suppose, at time t, given $p_t :=$ prediction of optimal algorithm's state o_t

Prediction error $\eta = \sum_t \text{dist}(p_t, o_t)$

There is an $O(\min\{\eta/\text{OPT}, \alpha\})$-competitive algorithm

Further, for caching, an $O(\min\{\log(\eta/\text{OPT}), \log k\})$-competitive algorithm

CONCERN

“Are your predictions plausible?” (anonymous reviewers)

CONCERN

“Are your predictions plausible?” (anonymous reviewers)

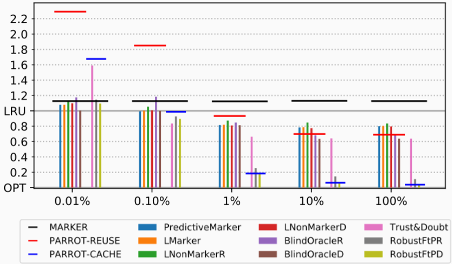
Yes! (Chłędowski, **P.**, Szabucki, Żołna, ICML 2021)

CONCERN

"Are your predictions plausible?" (anonymous reviewers)

Yes! (Chłędowski, **P.**, Szabucki, Żołna, ICML 2021)

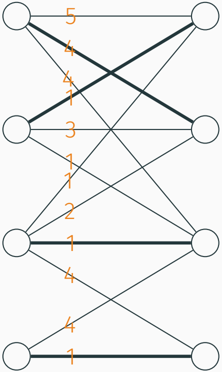
…and algorithms using them tend to perform better



We used **Parrot** (Liu et al., ICML 2020), a neural network for caching problem, with two heads:

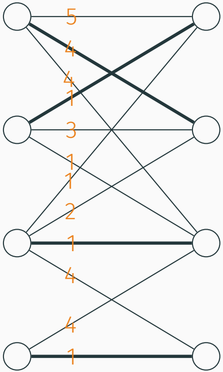
- next arrival time
- item evicted by optimal algorithm

MINIMUM WEIGHT BIPARTITE MATCHING



Hungarian algorithm: $O(nm)$ time

MINIMUM WEIGHT BIPARTITE MATCHING



Hungarian algorithm: $O(nm)$ time

What if we solve many similar instances? E.g.:

- instances sampled from a distribution,
- one instance slowly changing over time.

RECALL: LP FORMULATION OF MATCHING

Primal:

$$
\begin{aligned}
\text{minimize} \quad & \sum_{e \in E} c_e x_e \\
\text{subject to} \quad & \sum_{e \in N(v)} x_e = 1 \quad && \forall\, v \in V \\
& x_e \geqslant 0 && \forall\, e \in E
\end{aligned}
$$

Dual:

$$
\begin{aligned}
\text{maximize} \quad & \sum_{v \in V} y_v \\
\text{subject to} \quad & y_u + y_v \leqslant c_{u,v} \quad && \forall\, (u,v) \in E
\end{aligned}
$$

Theorem: (Dinitz et al., arXiv 2021)

Suppose input comes with **predicted dual** $\hat{y}$

There is an $O(m\sqrt{n} \cdot \min\{||\hat{y} - y||_1, \sqrt{n}\})$-time algorithm

OPEN PROBLEMS

Use predictions to speed-up other combinatorial optimization algorithms

- Minimum cost maximum flow
- Local search algorithms
- Put your favorite algorithm here

OPEN PROBLEMS

Use predictions to speed-up other combinatorial optimization algorithms

- Minimum cost maximum flow
- Local search algorithms
- Put your favorite algorithm here

Thank you!