Online Scheduling via Learned Weights

Silvio Lattanzi, **Thomas Lavastida**,
Benjamin Moseley, Sergei Vassilvitskii
8/2/2021

ML is Everywhere

- Massive successes in image classification, NLP, etc.
- Widening array of applications
- Can ML help to improve algorithms for combinatorial problems?

Algorithms with ML

- Find a good algorithm for "practical" instances?
  - Hard to characterize these
  - Usually resort to worst case analysis or stochastic analysis

- What if we have data? E.g., past instances

Algorithms with ML

- Learn from instances of the problem
- Design a good algorithm using predictions from past data, approaching "instance-optimal"
- Need to handle errors

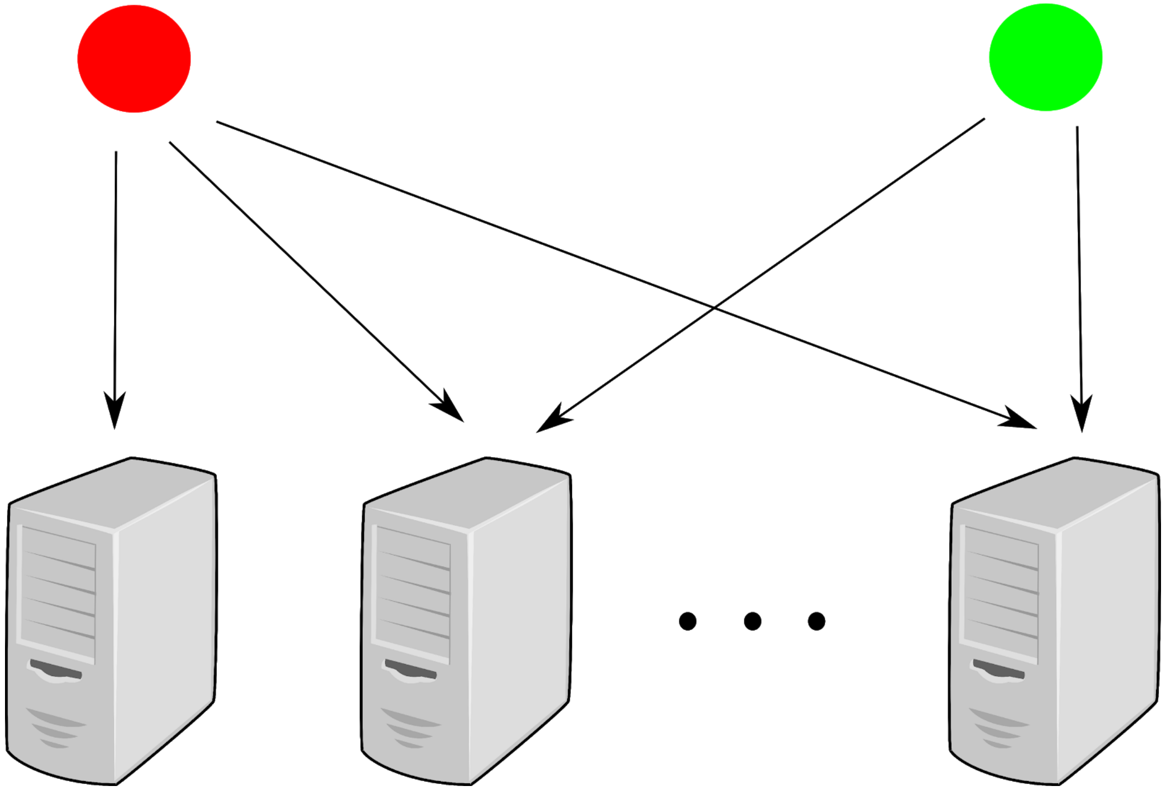
Algorithms with ML

- Caching Problem [Lykouris and Vassilvitskii 2018]
- Ski Rental + Non-Preemptive Sched. [Purohit et al. 2018]
- Heavy Hitters Sketches [Hsu et al 2019]
- Improved Bloom Filters [Mitzenmacher 2018]
- Learned Index Structures [Kraska et al 2018]
- Metrical Task Systems [Antoniadis et al 2020]

… even more recently

Online Load Balancing

- m machines
- n jobs arrive in online list
  - Restricted assignments
  - $N(j) =$ subset of feasible machines for job j
  - $p_j =$ size of job j
- Machine load: total size of jobs assigned to a machine
- Goal: minimize makespan



Worst Case Analysis

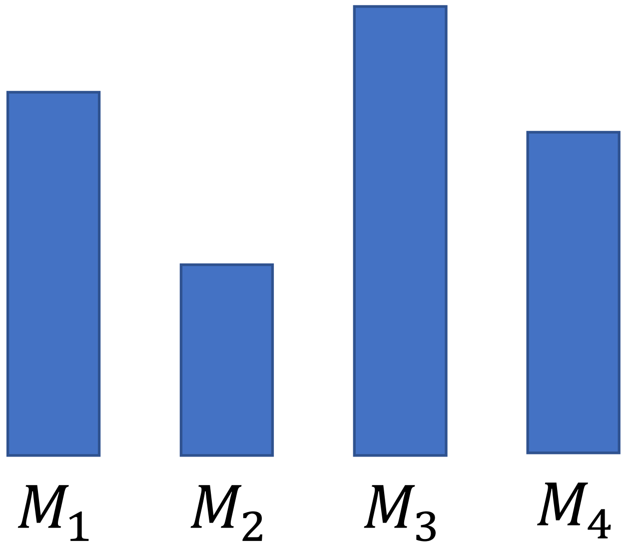
- Online algorithm c-competitive if for all inputs

$$ALG \leq c \cdot OPT$$

- Every algorithm $\Omega(\log m)$-competitive
- Greedy algorithm $O(\log m)$-competitive
  - [Azar, Naor, Rom 1995]

Predictions for Load Balancing?

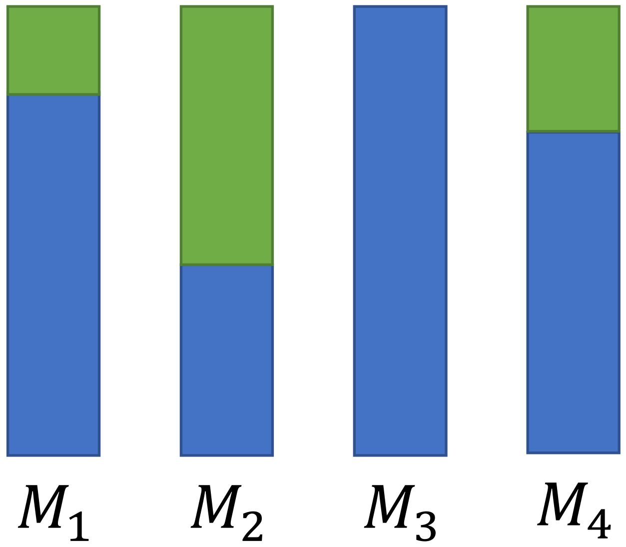
- Load of machines in OPT?
  - Pad the instance
- Dual variables?
  - Too sensitive to small errors
- Distribution over job subsets?
  - Potentially too many!



- Goal: Capture machine contentiousness

Predictions for Load Balancing?

- Load of machines in OPT?
  - Pad the instance
- Dual variables?
  - Too sensitive to small errors
- Distribution over job subsets?
  - Potentially too many!



- Goal: Capture machine contentiousness

Machine Weights

- Predict a single weight for each machine
- Lower weight corresponds to more contentious machines

- Framework:
  - "Correct" weights define a near optimal fractional assignment
  - Handle errors when the weights are predicted
  - Round to an integral solution online

Existence of Weights

<u>Theorem 1</u>

For any (offline) instance and $\epsilon > 0$ there exists weights $w \in R_+^m$ and a fractional assignment $x(w)$ with fractional makespan at most $(1+\epsilon)OPT$

- Machine i has weight w_i
- Fractional assignment: $x_{ij}(w) = \frac{w_i}{\sum_{i' \in N(j)} w_{i'}}$
- Want to satisfy $\sum_j p_j x_{ij}(w) \leq (1+\epsilon)OPT$ for all i
- Proof builds off [Agrawal et al. 2018]

Existence of Good Weights

- Initially all $w_i = 1$
- For some number of rounds R
  - Compute assignment using weights
  - Fractional load $L_i = \sum_j p_j x_{ij}(w)$
  - For all machines with $L_i \geq (1+\epsilon)OPT$
    - Set $w_i \leftarrow \frac{w_i}{1+\epsilon}$

Using Weights Online

Theorem 2

Given predicted weights w' there is an online algorithm yielding fractional assignments with makespan $O(\log \eta \, OPT)$

where $\eta := \max_i \frac{w_i'}{w_i}$ is the worst relative error in the predictions

Using Weights Online

- Given predicted weights w'
- Machines operate in phases
- Assign using current weights
- Update $w_i' \leftarrow \frac{w_i'}{2}$ if machine i gets load $10 \cdot OPT$
- Reset load and start a new phase
- If $\eta = \max_i \frac{w_i'}{w_i}$ then get $O(\log \eta)$-competitive assignment

Online Rounding Problem

- Receive j's size, neighborhood, fractional assignment online

$$\{x_{ij}\}_{i \in N(j)} \; s.t. \sum_{i \in N(j)} x_{ij} = 1$$

- Use x_{ij}'s to compute integral assignment online
- Rounding algorithm c-competitive if

$$ALG \leq c \cdot T$$

- $T \coloneqq \max\{\max_i \sum_j p_j x_{ij}, \max_j p_j\}$

Rounding Online

Theorem 3

There exists a randomized online rounding algorithm for restricted assignment which is $O((\log\log m)^3)$-competitive with high probability

Theorem 4

Any randomized online rounding algorithm is at least $\Omega\left(\frac{\log\log m}{\log\log\log m}\right)$-competitive

Conclusion

- Online fractional assignment + rounding yields $O((\log\log m)^3 \log\eta)$-competitive algorithm with predictions
- Moderately accurate predictions go beyond worst case
- Can retain $O(\log m)$-competitiveness when η large
  - Application of Mahdian et al. 2012
- Follow up work - weights are formally learnable
  - https://arxiv.org/abs/2011.11743
  - To appear in ESA 2021

Thank you!

Questions?