

Online Metric Algorithms with Untrusted Predictions

Antonios Antoniadis¹ **Christian Coester**² Marek Elias³
Adam Polak⁴ Bertrand Simon⁵

- 1: MPI, Saarbrücken (Germany).
- 2: CWI, Amsterdam (Netherlands).
- 3: EPFL, Lausanne (Switzerland).
- 4: Jagiellonian University, Kraków (Poland).
- 5: University of Bremen (Germany).

Online algorithms

- ▶ Optimization with incomplete information
- ▶ Guaranteed **competitive ratio**:
 r s.t. $\forall I: \text{ALG}(I) \leq r \cdot \text{OPT}(I)$
- ▶ Bad performance on easy instances, **overly pessimistic**



Motivation

Online algorithms

- ▶ Optimization with incomplete information
- ▶ Guaranteed **competitive ratio**:
 r s.t. $\forall I: \text{ALG}(I) \leq r \cdot \text{OPT}(I)$
- ▶ Bad performance on easy instances, **overly pessimistic**



Motivation

Online algorithms

- ▶ Optimization with incomplete information
- ▶ Guaranteed **competitive ratio**: r s.t. $\forall I: \text{ALG}(I) \leq r \cdot \text{OPT}(I)$
- ▶ Bad performance on easy instances, **overly pessimistic**



Machine learning predictions

- ▶ Often very powerful



- ▶ **No guarantee**, can be arbitrarily bad

Motivation

Online algorithms

- ▶ Optimization with incomplete information
- ▶ Guaranteed **competitive ratio**: r s.t. $\forall I: \text{ALG}(I) \leq r \cdot \text{OPT}(I)$
- ▶ Bad performance on easy instances, **overly pessimistic**



Machine learning predictions

- ▶ Often very powerful



- ▶ **No guarantee**, can be arbitrarily bad
- ▶ prediction error η

Motivation

Online algorithms

- ▶ Optimization with incomplete information
- ▶ Guaranteed competitive ratio:
 $r \text{ s.t. } \forall I: \text{ALG}(I) \leq r \cdot \text{OPT}(I)$
- ▶ Bad performance on easy instances, overly pessimistic



Machine learning predictions

- ▶ Often very powerful



- ▶ No guarantee, can be arbitrarily bad
- ▶ prediction error η

Prediction-augmented algorithms

- ▶ Target competitive ratio: $O(\min\{\text{ONLINE}, f(\eta/\text{OPT})\})$

Some previously studied problems

- ▶ Ski rental: predict #days we will ski [PurohitSK'18]
- ▶ Non-clairvoyant scheduling: predict processing times [PurohitSK'18]
- ▶ Restricted assignment: predict machine weights [LattanziLMV'20]
- ▶ Caching: predict next arrival time [LykourisV'18,Rohatgi'20,Wei'20]
- ▶ Weighted caching: predict *all* requests until next arrival [JiangPS'20]

Issue: lack of generality, predictions tailored to specific problems

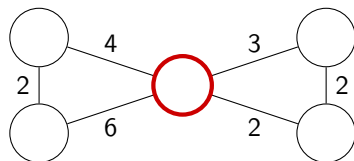
Proposition (Antoniadis, Coester, Elias, Polak, Simon; ICML'20)

*Previously used caching predictions **not** useful for weighted caching.*

⇒ need *more general* prediction setup

Metrical Task Systems (MTS)

- ▶ Given: metric space (M, d) , $x_0 \in M$
- ▶ At time t :
 1. cost function $\ell_t : M \mapsto \mathbb{R}_+$ revealed
 2. algo chooses $x_t \in M$
 3. pays $d(x_{t-1}, x_t) + \ell_t(x_t)$

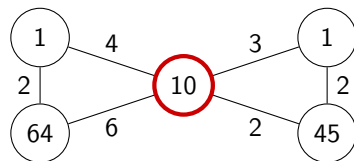


Cost: 0

MTS generalizes caching, k -server, convex body/function chasing, layered graph traversal...

Metrical Task Systems (MTS)

- ▶ Given: metric space (M, d) , $x_0 \in M$
- ▶ At time t :
 1. cost function $\ell_t : M \mapsto \mathbb{R}_+$ revealed
 2. algo chooses $x_t \in M$
 3. pays $d(x_{t-1}, x_t) + \ell_t(x_t)$

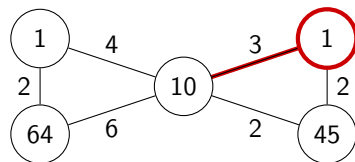


Cost: 0

MTS generalizes caching, k -server, convex body/function chasing, layered graph traversal...

Metrical Task Systems (MTS)

- ▶ Given: metric space (M, d) , $x_0 \in M$
- ▶ At time t :
 1. cost function $\ell_t : M \mapsto \mathbb{R}_+$ revealed
 2. algo chooses $x_t \in M$
 3. pays $d(x_{t-1}, x_t) + \ell_t(x_t)$

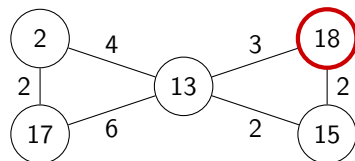


Cost: 3+1

MTS generalizes caching, k -server, convex body/function chasing, layered graph traversal...

Metrical Task Systems (MTS)

- ▶ Given: metric space (M, d) , $x_0 \in M$
- ▶ At time t :
 1. cost function $\ell_t : M \mapsto \mathbb{R}_+$ revealed
 2. algo chooses $x_t \in M$
 3. pays $d(x_{t-1}, x_t) + \ell_t(x_t)$

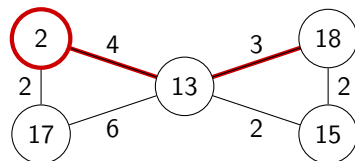


Cost: $3+1$

MTS generalizes caching, k -server, convex body/function chasing, layered graph traversal...

Metrical Task Systems (MTS)

- ▶ Given: metric space (M, d) , $x_0 \in M$
- ▶ At time t :
 1. cost function $\ell_t : M \mapsto \mathbb{R}_+$ revealed
 2. algo chooses $x_t \in M$
 3. pays $d(x_{t-1}, x_t) + \ell_t(x_t)$

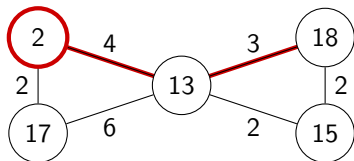


Cost: $3+1+7+2$

MTS generalizes caching, k -server, convex body/function chasing, layered graph traversal...

Metric Task Systems (MTS)

- ▶ Given: metric space (M, d) , $x_0 \in M$
- ▶ At time t :
 1. cost function $\ell_t : M \mapsto \mathbb{R}_+$ revealed
 2. algo chooses $x_t \in M$
 3. pays $d(x_{t-1}, x_t) + \ell_t(x_t)$



Cost: $3+1+7+2$

MTS generalizes caching, k -server, convex body/function chasing, layered graph traversal...

What to predict?

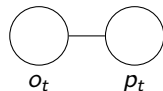
- ▶ Next costs?
 - only ℓ_{t+1} ? useless with dummy rounds
 - entire future? too much information
- ▶ **Single state per round**: recommendation for x_t

Definition of the error and a simple algorithm

Definition: error η

- ▶ Fix OFF: offline algorithm (e.g., OPT), who goes to states $o_1, o_2, \dots$
- ▶ At time t , $p_t :=$ prediction of o_t .
- ▶ Error: $\eta := \sum_t d(o_t, p_t)$

Goal: Small cost (rel. to OFF) if η small



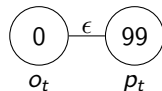
Definition of the error and a simple algorithm

Definition: error η

- ▶ Fix OFF: offline algorithm (e.g., OPT), who goes to states $o_1, o_2, \dots$
- ▶ At time t , $p_t :=$ prediction of o_t .
- ▶ Error: $\eta := \sum_t d(o_t, p_t)$

Goal: Small cost (rel. to OFF) if η small

- ▶ Naive algo: $x_t := p_t$



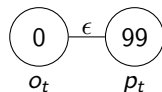
Definition of the error and a simple algorithm

Definition: error η

- ▶ Fix OFF: offline algorithm (e.g., OPT), who goes to states $o_1, o_2, \dots$
- ▶ At time t , $p_t :=$ prediction of o_t .
- ▶ Error: $\eta := \sum_t d(o_t, p_t)$

Goal: Small cost (rel. to OFF) if η small

- ▶ Naive algo: $x_t := p_t$
- ▶ **Better:** $x_t := \arg \min_{x \in X} \{cost_t(x) + 2d(p_t, x)\}$
- ▶ Call this algo F_{TP} (Follow the Prediction)



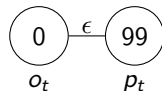
Definition of the error and a simple algorithm

Definition: error η

- ▶ Fix OFF: offline algorithm (e.g., OPT), who goes to states $o_1, o_2, \dots$
- ▶ At time t , $p_t :=$ prediction of o_t .
- ▶ Error: $\eta := \sum_t d(o_t, p_t)$

Goal: Small cost (rel. to OFF) if η small

- ▶ Naive algo: $x_t := p_t$
- ▶ **Better:** $x_t := \arg \min_{x \in X} \{cost_t(x) + 2d(p_t, x)\}$
- ▶ Call this algo FTP (Follow the Prediction)



Lemma (Antoniadis, Coester, Elias, Polak, Simon; ICML'20)

FTP has cost $\leq \text{OFF} + 4\eta$. So competitive ratio vs OFF is $1 + 4\eta/\text{OFF}$.

Guarantee holds simultaneously for **all** offline algos (η depending on it)

Issue: FTP not robust

Combine online algorithms **A** and **B**: $\text{comb}(A, B)$ [BlumB'00]

$$\blacktriangleright E(\text{cost}_{\text{comb}(A,B)}) \leq (1+\varepsilon) \cdot \min\{E(\text{cost}_A), E(\text{cost}_B)\} + O(\text{diameter}/\varepsilon)$$

Let $\text{ROBUSTFTP} := \text{comb}(\text{ONLINE}, \text{FTP})$

Combine online algorithms **A** and **B**: $comb(A, B)$ [BlumB'00]

- ▶ $E(cost_{comb(A,B)}) \leq (1+\varepsilon) \cdot \min\{E(cost_A), E(cost_B)\} + O(\text{diameter}/\varepsilon)$

Let $ROBUSTFTP := comb(ONLINE, FTP)$

Theorem (Antoniadis, Coester, Elias, Polak, Simon; ICML'20)

$ROBUSTFTP$ has cost $O(\min\{\text{OFF} + \eta, ONLINE\})$.

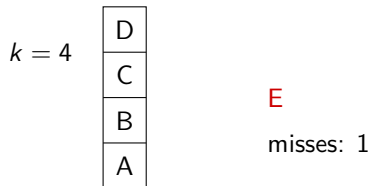
(Recall: $\eta = \text{prediction error wrt. OFF}$)

This is asymptotically optimal.

- ▶ Lower bound holds for **some** MTS
- ▶ For caching (a special case of MTS), we can do better

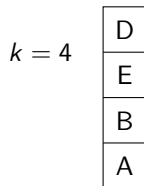
Caching (aka Paging)

- ▶ Maintain a cache of k pages, pay 1 per cache miss



Caching (aka Paging)

- ▶ Maintain a cache of k pages, pay 1 per cache miss

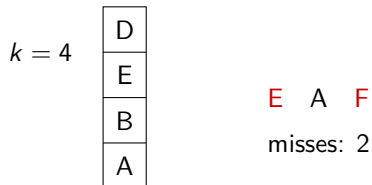


E A

misses: 1

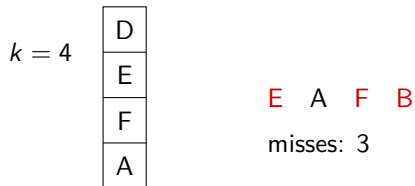
Caching (aka Paging)

- Maintain a cache of k pages, pay 1 per cache miss



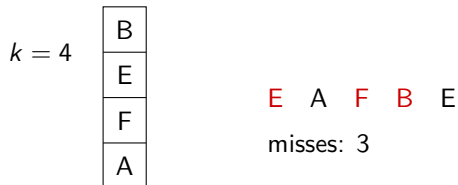
Caching (aka Paging)

- ▶ Maintain a cache of k pages, pay 1 per cache miss



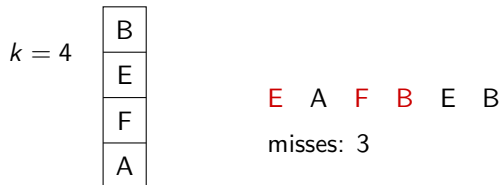
Caching (aka Paging)

- ▶ Maintain a cache of k pages, pay 1 per cache miss



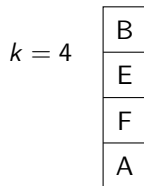
Caching (aka Paging)

- Maintain a cache of k pages, pay 1 per cache miss



Caching (aka Paging)

- ▶ Maintain a cache of k pages, pay 1 per cache miss

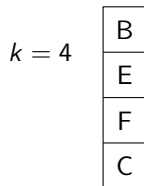


E A F B E B C

misses: 4

Caching (aka Paging)

- ▶ Maintain a cache of k pages, pay 1 per cache miss



E A F B E B C F

misses: 4

Caching (aka Paging)

- ▶ Maintain a cache of k pages, pay 1 per cache miss



- ▶ Predicted state: set of k pages
(intuition: recommended cache content)

Algorithm TRUST&DOUBT: idea

- ▶ Balance 2 competing policies:
 - “Trust”: Evict what is evicted from predicted cache
 - “Doubt”: Evict according to classical policy
- ▶ In a phase, first follow “Trust”
- ▶ If this turns out bad, switch to “Doubt”
- ▶ Regularly (depending on trustworthiness): “Trust” again

Theorem (Antoniadis, Coester, Elias, Polak, Simon; ICML'20)

TRUST&DOUBT is $\min \left\{ O(\log k), O(\log \frac{\eta}{O_{PT}}) \right\}$ -competitive.

Algorithm TRUST&DOUBT: idea

- ▶ Balance 2 competing policies:
 - “Trust”: Evict what is evicted from predicted cache
 - “Doubt”: Evict according to classical policy
- ▶ In a phase, first follow “Trust”
- ▶ If this turns out bad, switch to “Doubt”
- ▶ Regularly (depending on trustworthiness): “Trust” again

Theorem (Antoniadis, Coester, Elias, Polak, Simon; ICML'20)

TRUST&DOUBT is $\min \left\{ O(\log k), O(\log \frac{\eta}{\text{OPT}}) \right\}$ -competitive.

Algorithm TRUST&DOUBT: idea

- ▶ Balance 2 competing policies:
 - “Trust”: Evict what is evicted from predicted cache
 - “Doubt”: Evict according to classical policy
- ▶ In a phase, first follow “Trust”
- ▶ If this turns out bad, switch to “Doubt”
- ▶ Regularly (depending on trustworthiness): “Trust” again

Theorem (Antoniadis, Coester, Elias, Polak, Simon; ICML'20)

TRUST&DOUBT is $\min \left\{ O(\log k), O(\log \frac{\eta}{\text{OFF}}) \right\}$ -competitive vs OFF.

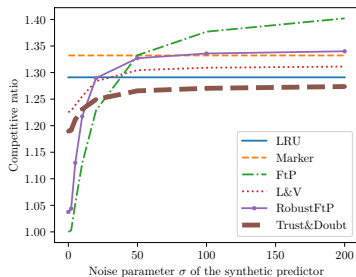
Comparison with previous prediction setup for caching

- ▶ Our setup: Prediction = recommended cache content
- ▶ In [LykourisV'18, Rohatgi'20, Wei'20]: Prediction = next time of each request

How do setups/results compare?

- ▶ Can **convert** request time predictions to cache content predictions
 - Predictor evicts furthest predicted page
- ▶ Prediction error η incomparable
 - Nonetheless: **TRUST&DOUBT** also achieves guarantee of [Wei'20]
- ▶ **Succinct**: Instead of $\log T$ bit predictions per request, only $\log k$ bits when predictor has cache miss
- ▶ Our setup **generalizes** to MTS

Same datasets as [LykourisV'18]



Prediction: ground truth + lognorm error

Dataset	BK	Citi		
LRU	1.29	1.85		
MARKER	1.33	1.86		
Predictions	PLECO	POPU	PLECO	POPU
L&V [LykourisV'18]	1.34	1.26	1.88	1.78
LMarker [Rohatgi'20]	1.34	1.26	1.88	1.78
LNonMarker [Rohatgi'20]	1.34	1.29	1.88	1.80
ROBUSTFTP	1.35	1.32	1.89	1.83
TRUST&DOUBT	1.29	1.27	1.85	1.77

Two predictors: simple statistics

Beyond MTS: online matching on the line

Definition by picture: n servers; n online requests



Analogous prediction setup

- ▶ Predict set of servers to be matched so far
- ▶ Error: cost of matching between these and OFF's matched servers

Theorem (Antoniadis, Coester, Elias, Polak, Simon; ICML'20)

ROBUSTFTP's analogue is $O(\min\{1 + \frac{\eta}{\text{OFF}}, \log n\})$ -competitive vs OFF.

Beyond MTS: online matching on the line

Definition by picture: n servers; n online requests



Analogous prediction setup

- ▶ Predict set of servers to be matched so far
- ▶ Error: cost of matching between these and OFF's matched servers

Theorem (Antoniadis, Coester, Elias, Polak, Simon; ICML'20)

ROBUSTFTP's analogue is $O(\min\{1 + \frac{\eta}{\text{OFF}}, \log n\})$ -competitive vs OFF.

Beyond MTS: online matching on the line

Definition by picture: n servers; n online requests



Analogous prediction setup

- ▶ Predict set of servers to be matched so far
- ▶ Error: cost of matching between these and OFF's matched servers

Theorem (Antoniadis, Coester, Elias, Polak, Simon; ICML'20)

ROBUSTFTP's analogue is $O(\min\{1 + \frac{\eta}{\text{OFF}}, \log n\})$ -competitive vs OFF.

Beyond MTS: online matching on the line

Definition by picture: n servers; n online requests



Analogous prediction setup

- ▶ Predict set of servers to be matched so far
- ▶ Error: cost of matching between these and OFF's matched servers

Theorem (Antoniadis, Coester, Elias, Polak, Simon; ICML'20)

ROBUSTFTP's analogue is $O(\min\{1 + \frac{\eta}{\text{OFF}}, \log n\})$ -competitive vs OFF.

Beyond MTS: online matching on the line

Definition by picture: n servers; n online requests



Analogous prediction setup

- ▶ Predict set of servers to be matched so far
- ▶ Error: cost of matching between these and OFF's matched servers

Theorem (Antoniadis, Coester, Elias, Polak, Simon; ICML'20)

ROBUSTFTP's analogue is $O(\min\{1 + \frac{\eta}{\text{OFF}}, \log n\})$ -competitive vs OFF.

Beyond MTS: online matching on the line

Definition by picture: n servers; n online requests



Analogous prediction setup

- ▶ Predict set of servers to be matched so far
- ▶ Error: cost of matching between these and OFF's matched servers

Theorem (Antoniadis, Coester, Elias, Polak, Simon; ICML'20)

ROBUSTFTP's analogue is $O(\min\{1 + \frac{\eta}{\text{OFF}}, \log n\})$ -competitive vs OFF.

Beyond MTS: online matching on the line

Definition by picture: n servers; n online requests



Analogous prediction setup

- ▶ Predict set of servers to be matched so far
- ▶ Error: cost of matching between these and OFF's matched servers

Theorem (Antoniadis, Coester, Elias, Polak, Simon; ICML'20)

ROBUSTFTP's analogue is $O(\min\{1 + \frac{\eta}{\text{OFF}}, \log n\})$ -competitive vs OFF.

Beyond MTS: online matching on the line

Definition by picture: n servers; n online requests



Analogous prediction setup

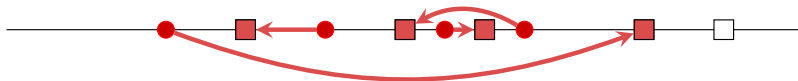
- ▶ Predict set of servers to be matched so far
- ▶ Error: cost of matching between these and OFF's matched servers

Theorem (Antoniadis, Coester, Elias, Polak, Simon; ICML'20)

ROBUSTFTP's analogue is $O(\min\{1 + \frac{\eta}{\text{OFF}}, \log n\})$ -competitive vs OFF.

Beyond MTS: online matching on the line

Definition by picture: n servers; n online requests



Analogous prediction setup

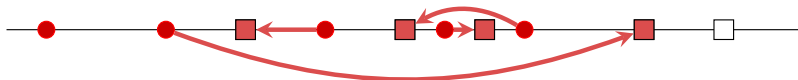
- ▶ Predict set of servers to be matched so far
- ▶ Error: cost of matching between these and OFF's matched servers

Theorem (Antoniadis, Coester, Elias, Polak, Simon; ICML'20)

ROBUSTFTP's analogue is $O(\min\{1 + \frac{\eta}{\text{OFF}}, \log n\})$ -competitive vs OFF.

Beyond MTS: online matching on the line

Definition by picture: n servers; n online requests



Analogous prediction setup

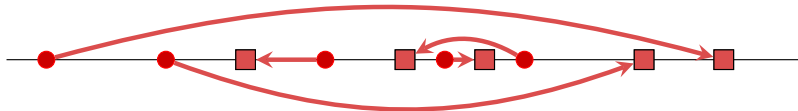
- ▶ Predict set of servers to be matched so far
- ▶ Error: cost of matching between these and OFF's matched servers

Theorem (Antoniadis, Coester, Elias, Polak, Simon; ICML'20)

ROBUSTFTP's analogue is $O(\min\{1 + \frac{\eta}{\text{OFF}}, \log n\})$ -competitive vs OFF.

Beyond MTS: online matching on the line

Definition by picture: n servers; n online requests



Analogue prediction setup

- ▶ Predict set of servers to be matched so far
- ▶ Error: cost of matching between these and OFF's matched servers

Theorem (Antoniadis, Coester, Elias, Polak, Simon; ICML'20)

ROBUSTFTP's analogue is $O(\min\{1 + \frac{\eta}{\text{OFF}}, \log n\})$ -competitive vs OFF.



MTS (and beyond)

- ▶ General prediction setup
- ▶ “Optimal” algorithm: `ROBUSTFTP`

Caching

- ▶ Better algorithm: `TRUST&DOUBT`
- ▶ Less predicted information and better empirical performance than caching-specific setup
- ▶ Better than LRU with simple predictor

Perspectives

- ▶ Better algorithms for other MTS?
(e.g., weighted caching, k -server, convex body chasing, graph traversal)