Online Metric Algorithms with Untrusted Predictions

Antonios Antoniadis[1] Christian Coester[2] Marek Elias[3]
Adam Polak[4] **Bertrand Simon**[5]

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

Grenoble – March 2020

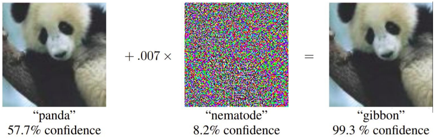
Motivation

Online algorithms

- Guaranteed competitive ratio $= \max_I \; ALG(I) \,/\, OPT(I)$
- Bad performance on easy instances, overly pessimistic
- *Stochastic algorithms: unrealistic*

Machine Learning predictions

- Often relevant information
- No guarantee, can be arbitrarily bad



arxiv.org/abs/1412.6572

Challenges

Improve prediction generation

Use predictions in robust algorithms

Motivation

Online algorithms

- Guaranteed competitive ratio $= \max_I \; ALG(I) \,/\, OPT(I)$
- Bad performance on easy instances, overly pessimistic
- *Stochastic algorithms: unrealistic*

Machine Learning predictions

- Often relevant information
- No guarantee, can be arbitrarily bad

absolute error η

$\eta = \sum_t \eta_t$

Challenges

Improve prediction generation

Use predictions in robust algorithms

Related work

Prediction-augmented algorithms

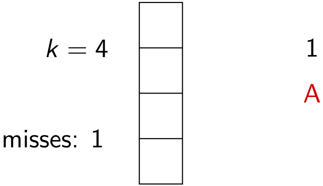
- Target competitive ratio: $O(\ \min\{\ 1 + f(\eta/\textsc{Opt})\ ,\ \mathit{ONLINE}\ \}\ )$

Some previously studied problems

- Ski rental: predict #days we will ski [KumarPS'18]
  Competitive ratio: e.g., $\min\{1.5 + 2\frac{\eta}{\textsc{Opt}}\ ,\ 3\}$
- Non-clairvoyant scheduling: predict processing times [KumarPS'18]
- Restricted assignment: predict machine weights [LattanziMLV'20]
- Caching: predict next arrival time [LykourisV'18, Rohatgi'20]
- ...

Issue: lack of generality, predictions tailored to specific problems

Caching with predictions [LykourisV'18, Rohatgi'20]



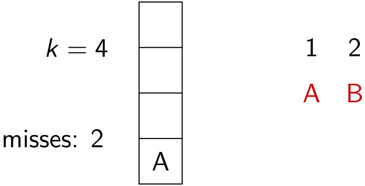
$k = 4$

misses: 1

1

A

Caching with predictions [LykourisV'18, Rohatgi'20]



$k = 4$

misses: 2

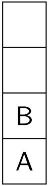
A

1	2
A	B

Caching with predictions [LykourisV'18, Rohatgi'20]

$k = 4$

misses: 2

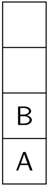


1	2	3
A	B	A

Caching with predictions [LykourisV'18, Rohatgi'20]

$k = 4$

misses: 3



B

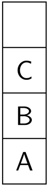
A

1	2	3	4
A	B	A	C

Caching with predictions [LykourisV'18, Rohatgi'20]

$k = 4$

misses: 4

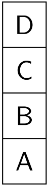


1	2	3	4	5
A	B	A	C	D

Caching with predictions [LykourisV'18, Rohatgi'20]

$k = 4$

misses: 5



D

C

B

A

1	2	3	4	5	6
A	B	A	C	D	E

Caching with predictions [LykourisV'18, Rohatgi'20]

$k = 4$

D
C
E
A

misses: 6

1	2	3	4	5	6	7
A	B	A	C	D	E	F

Caching with predictions [LykourisV'18, Rohatgi'20]

$k = 4$

misses: 6

D
F
E
A

1	2	3	4	5	6	7	8
A	B	A	C	D	E	F	A

Caching with predictions [LykourisV'18, Rohatgi'20]

$k = 4$

misses: 7

D
F
E
A

1	2	3	4	5	6	7	8	9
A	B	A	C	D	E	F	A	B

Caching with predictions [LykourisV'18, Rohatgi'20]

$k = 4$

misses: 7

D
B
E
A

1	2	3	4	5	6	7	8	9	10
A	B	A	C	D	E	F	A	B	E

Caching with predictions [LykourisV'18, Rohatgi'20]

$k = 4$

misses: 8

D
B
E
A

1	2	3	4	5	6	7	8	9	10	11
A	B	A	C	D	E	F	A	B	E	F

Caching with predictions [LykourisV'18, Rohatgi'20]

$k = 4$

misses: 8

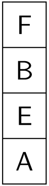
F
B
E
A

1	2	3	4	5	6	7	8	9	10	11
A	B	A	C	D	E	F	A	B	E	F

Caching with predictions [LykourisV'18, Rohatgi'20]

$k = 4$

misses: 8


F
B
E
A

	1	2	3	4	5	6	7	8	9	10	11
	A	B	A	C	D	E	F	A	B	E	F
next:	3	9	8	12	12	10	11	12	12	12	12

Caching with predictions [LykourisV'18, Rohatgi'20]

$k = 4$

F
B
E
A

misses: 8

	1	2	3	4	5	6	7	8	9	10	11
	A	B	A	C	D	E	F	A	B	E	F
next:	3	9	8	12	12	10	11	12	12	12	12

Predict next arrival times ($\eta = \sum |\text{predicted} - \text{truth}|$)

- Consider the online Marker algorithm, $O(\log k)$-competitive
- Sometimes, evict the furthest predicted page instead of random
- Competitive ratios: $O(\min\{\log k\ ,\ 1 + \sqrt{\frac{\eta}{\textsc{Opt}}}\})$

Lemma (Antoniadis, Coester, Elias, Polak, **S.**)

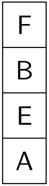
These predictions are not useful for weighted caching.

$\Longrightarrow$ need for a *more general* prediction setup

Caching with predictions [LykourisV'18, Rohatgi'20]

$k = 4$

misses: 8



	1	2	3	4	5	6	7	8	9	10	11
	A	B	A	C	D	E	F	A	B	E	F
next:	3	9	8	12	12	10	11	12	12	12	12

Predict next arrival times ($\eta = \sum |\text{predicted} - \text{truth}|$)

- Consider the online Marker algorithm, $O(\log k)$-competitive
- Sometimes, evict the furthest predicted page instead of random
- Competitive ratios: $O(\min\{\log k\ ,\ 1 + \log \frac{\eta}{\textsc{Opt}}\})$

Lemma (Antoniadis, Coester, Elias, Polak, S.)

These predictions are not useful for weighted caching.

$\Longrightarrow$ need for a *more general* prediction setup

Caching with predictions [LykourisV'18, Rohatgi'20]

$k = 4$

misses: 8

F
B
E
A

	1	2	3	4	5	6	7	8	9	10	11
	A	B	A	C	D	E	F	A	B	E	F
next:	3	9	8	12	12	10	11	12	12	12	12

Predict next arrival times ($\eta = \sum |\text{predicted} - \text{truth}|$)

- Consider the online Marker algorithm, $O(\log k)$-competitive
- Sometimes, evict the furthest predicted page instead of random
- Competitive ratios: $O(\min\{\log k\ ,\ 1 + \frac{\eta}{\textsc{Opt}} \cdot \frac{\log k}{k}\})$

Lemma (Antoniadis, Coester, Elias, Polak, S.)

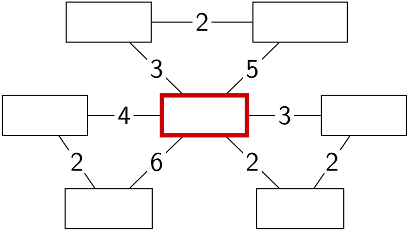
These predictions are not useful for weighted caching.

$\Longrightarrow$ need for a *more general* prediction setup

The Metrical Task System (MTS) problem

Definition by picture

Note: generalizes caching, *k*-server, convex body chasing...

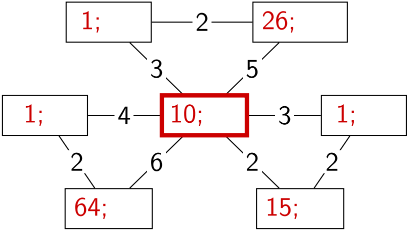


Round 0
Cost incurred: 0

The Metrical Task System (MTS) problem

Definition by picture

Note: generalizes caching, *k*-server, convex body chasing...

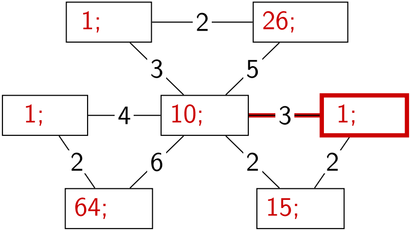


Round 1 before serving
Cost incurred: 0

The Metrical Task System (MTS) problem

Definition by picture

Note: generalizes caching, *k*-server, convex body chasing. . .

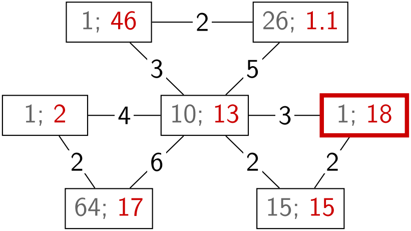


Round 1 after serving
Cost incurred: 3+1

The Metrical Task System (MTS) problem

Definition by picture

Note: generalizes caching, *k*-server, convex body chasing...

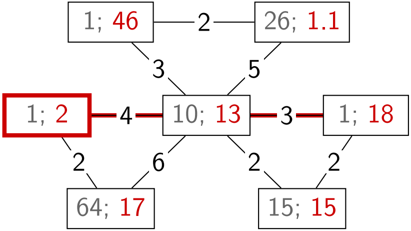


Round 2 before serving
Cost incurred: 3+1

The Metrical Task System (MTS) problem

Definition by picture

Note: generalizes caching, *k*-server, convex body chasing...

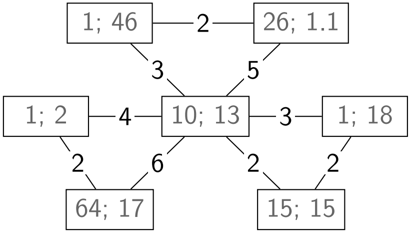


Round 2 after serving
Cost incurred: 3+1+7+2

The Metrical Task System (MTS) problem

Definition by picture

Note: generalizes caching, *k*-server, convex body chasing. . .



Round 2 serving

Cost incurred: 3+1+7+2

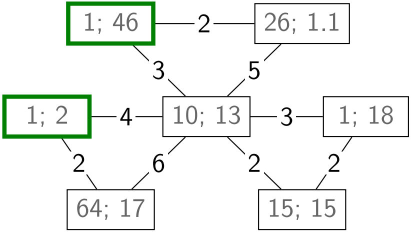
What should we predict?

- Next costs?
  - few rounds: useless with dummy rounds
  - many rounds: too much information
- Single state per round: where we should be
  - Distance VS optimal state? there can be several good options. . .

The Metrical Task System (MTS) problem

Definition by picture

Note: generalizes caching, *k*-server, convex body chasing…



Round 2 serving

Cost incurred: 3+1+7+2

What should we predict?

- Next costs?
  - few rounds: useless with dummy rounds
  - many rounds: too much information
- Single state per round: where we should be
  - Distance VS optimal state? there can be several good options…

Definition of the error and first algorithm: $\textsc{FtP}$

How to measure the error?

- Fix $\textsc{Off}$: offline algorithm (e.g., $\textsc{Opt}$), who goes to states $o_1, o_2, \ldots$
- At time t, $p_t :=$ prediction of o_t.
- Error: $\eta = \sum_t d(o_t, p_t)$

Algorithm subroutine ($\textsc{FtP}$, Follow the Prediction)

- Go to p_t, except if there is a cheap state nearby
  i.e., go to the state $\arg\min_{x \in X}\{cost_t(x) + 2d(x, p_t)\}$
- Proposition: this costs at most $\textsc{Off} \cdot (1 + 4\eta/\textsc{Off})$
  note that $cost_t(p_t)$ *is irrelevant*

Comments on $\textsc{Off}$

- The guarantee is true for any algorithm $\textsc{Off}$ (η depending on it)
- Intuition: $\textsc{FtP}$ is good if there is a good algorithm close to $\{p_t\}$

Issue: $\textsc{FtP}$ is not robust

Algorithm for MTS: RobustFtP

Combining online algorithms A and B: $comb(A, B)$ [BlumB'00] [FiatRR'94]

- $E(cost_{comb(A,B)}) \leq (1+\varepsilon) \cdot \min\{E(cost_A), E(cost_B)\} + O(\text{diameter}/\varepsilon)$
- Deterministic: $cost_{comb(A,B)} \leq 9 \cdot \min\{E(cost_A), E(cost_B)\}$

Theorem (RobustFtP := $comb(ONLINE, \text{FtP})$ competitive ratio)

RobustFtP *costs* $O(\min\{\text{Off} \cdot (1 + \eta/\text{Off}) \ , \ ONLINE\})$.

(Recall: $\eta = \sum_t d(o_t, p_t)$*)*

Lemma (Lower bound)

This is tight for some MTS (i.e., η/Off *- dependency).*

For a specific MTS (e.g., caching), the lower bound does not hold

Logarithmic error dependency for caching

Focus on a specific MTS: the caching problem

- Maintain a cache of k pages, pay 1 per cache miss
- Recall: prediction = cache of some offline algorithm $\textsc{Off}$

Algorithm $\textsc{Trust\&Doubt}$: sketch

- Phases as in the Marker algorithm (= k different requests)
- *Clean* page q arrives: "trust" the predictor for q; evict some page p_q not in the predictor's cache
- If a p_q is requested: "doubt" the predictor for q; pick another p_q (may be in our cache)
- Regularly (depending on trustworthiness): evict p_q, "trust" for q

Theorem ($\textsc{Trust\&Doubt}$ competitive ratio)

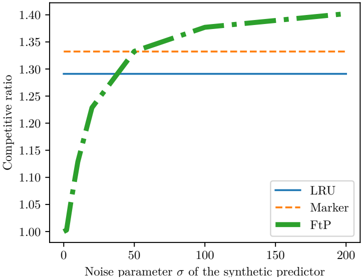
$\textsc{Trust\&Doubt}$ *costs* $O(\min\{\textsc{Off} \cdot (1 + \log \frac{\eta}{\textsc{Off}})\ ,\ \textsc{Opt} \cdot \log k\})$.

Comparison to previous caching algorithms

How do our algorithms compare to previous ones?

- Different errors → difficult to compare competitive ratios
- Experimentally: compute previous predictions
  deduct our predictions (evict furthest predicted)

Results on a public dataset (BrightKite, $k = 10$) – *(lower is better)*



Prediction: ground truth + lognorm error

LRU	1.291	
Marker	1.333	
Predictions	PLECO	POPU
FtP	2.081	1.707
L&V [LykourisV'18]	1.340	1.262
LMarker [Rohatgi'20]	1.337	1.264
LNonMarker [Rohatgi'20]	1.339	1.292
RobustFtP	**1.351**	**1.316**
Trust&Doubt	**1.292**	**1.274**

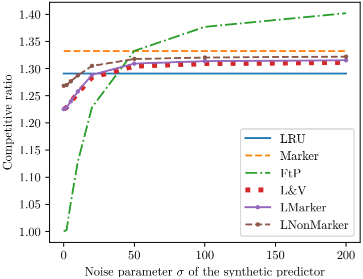
Two *predictors*: simple statistics

Comparison to previous caching algorithms

How do our algorithms compare to previous ones?

- Different errors → difficult to compare competitive ratios
- Experimentally: compute previous predictions
  deduct our predictions (evict furthest predicted)

Results on a public dataset (BrightKite, $k = 10$) – *(lower is better)*



Prediction: ground truth + lognorm error

LRU	1.291	
Marker	1.333	
Predictions	PLECO	POPU
FtP	2.081	1.707
L&V [LykourisV'18]	1.340	1.262
LMarker [Rohatgi'20]	1.337	1.264
LNonMarker [Rohatgi'20]	1.339	1.292
RobustFtP	**1.351**	**1.316**
Trust&Doubt	**1.292**	**1.274**

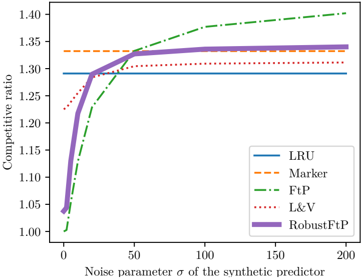
Two *predictors*: simple statistics

Comparison to previous caching algorithms

How do our algorithms compare to previous ones?

- Different errors → difficult to compare competitive ratios
- Experimentally: compute previous predictions
  deduct our predictions (evict furthest predicted)

Results on a public dataset (BrightKite, $k = 10$) – *(lower is better)*



Prediction: ground truth + lognorm error

LRU	1.291	
Marker	1.333	
Predictions	PLECO	POPU
FtP	2.081	1.707
L&V [LykourisV'18]	1.340	1.262
LMarker [Rohatgi'20]	1.337	1.264
LNonMarker [Rohatgi'20]	1.339	1.292
RobustFtP	**1.351**	**1.316**
Trust&Doubt	**1.292**	**1.274**

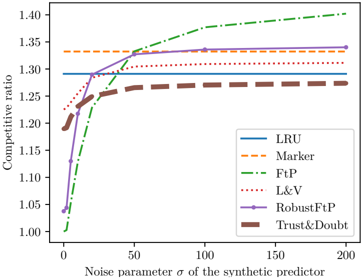
Two *predictors*: simple statistics

Comparison to previous caching algorithms

How do our algorithms compare to previous ones?

- Different errors → difficult to compare competitive ratios
- Experimentally: compute previous predictions
  deduct our predictions (evict furthest predicted)

Results on a public dataset (BrightKite, $k = 10$) – *(lower is better)*



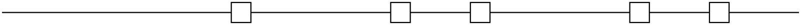
Prediction: ground truth + lognorm error

LRU	1.291	
Marker	1.333	
Predictions	PLECO	POPU
FTP	2.081	1.707
L&V [LykourisV'18]	1.340	1.262
LMarker [Rohatgi'20]	1.337	1.264
LNonMarker [Rohatgi'20]	1.339	1.292
ROBUSTFTP	**1.351**	**1.316**
TRUST&DOUBT	**1.292**	**1.274**

Two *predictors*: simple statistics

Beyond MTS: online matching on the line

Definition by picture: n servers; n online requests



Remarks

- This problem is not known to be an MTS
- Binding decisions: cannot pay to modify the matching
- Can *simulate* another matching

Analoguous prediction setup

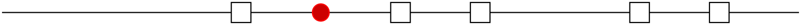
- Predict the set of severs matched in some $\textsc{Off}$
- Error: min cost of a perfect matching between these servers

Theorem

$\textsc{RobustFtP}$'s analoguous costs $O(\min\{\textsc{Off}\cdot(1+\frac{\eta}{\textsc{Off}})\ ,\ \textsc{Opt}\cdot\log n\})$.

Beyond MTS: online matching on the line

Definition by picture: *n* servers; *n* online requests



Remarks

- This problem is not known to be an MTS
- Binding decisions: cannot pay to modify the matching
- Can *simulate* another matching

Analoguous prediction setup

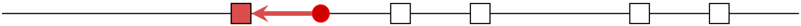
- Predict the set of severs matched in some $\textsc{Off}$
- Error: min cost of a perfect matching between these servers

Theorem

$\textsc{RobustFtP}$*'s analoguous costs* $O(\min\{\textsc{Off}\cdot(1+\frac{\eta}{\textsc{Off}})\ ,\ \textsc{Opt}\cdot\log n\})$.

Beyond MTS: online matching on the line

Definition by picture: n servers; n online requests



Remarks

- This problem is not known to be an MTS
- Binding decisions: cannot pay to modify the matching
- Can *simulate* another matching

Analoguous prediction setup

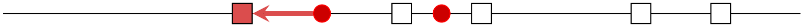
- Predict the set of severs matched in some $\textsc{Off}$
- Error: min cost of a perfect matching between these servers

Theorem

$\textsc{RobustFtP}$*'s analoguous costs* $O(\min\{\textsc{Off}\cdot(1+\frac{\eta}{\textsc{Off}})\ ,\ \textsc{Opt}\cdot\log n\})$.

Beyond MTS: online matching on the line

Definition by picture: n servers; n online requests



Remarks

- This problem is not known to be an MTS
- Binding decisions: cannot pay to modify the matching
- Can *simulate* another matching

Analoguous prediction setup

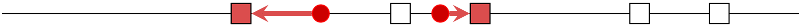
- Predict the set of severs matched in some $\textsc{Off}$
- Error: min cost of a perfect matching between these servers

Theorem

$\textsc{RobustFtP}$*'s analoguous costs* $O(\min\{\textsc{Off}\cdot(1+\frac{\eta}{\textsc{Off}})\ ,\ \textsc{Opt}\cdot\log n\})$.

Beyond MTS: online matching on the line

Definition by picture: n servers; n online requests



Remarks

- This problem is not known to be an MTS
- Binding decisions: cannot pay to modify the matching
- Can *simulate* another matching

Analoguous prediction setup

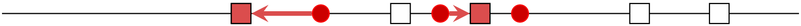
- Predict the set of severs matched in some $\textsc{Off}$
- Error: min cost of a perfect matching between these servers

Theorem

$\textsc{RobustFtP}$*'s analoguous costs* $O(\min\{\textsc{Off} \cdot (1 + \frac{\eta}{\textsc{Off}})\ ,\ \textsc{Opt} \cdot \log n\})$.

Beyond MTS: online matching on the line

Definition by picture: n servers; n online requests



Remarks

- This problem is not known to be an MTS
- Binding decisions: cannot pay to modify the matching
- Can *simulate* another matching

Analoguous prediction setup

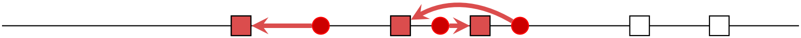
- Predict the set of severs matched in some $\textsc{Off}$
- Error: min cost of a perfect matching between these servers

Theorem

$\textsc{RobustFtP}$*'s analoguous costs* $O(\min\{\textsc{Off} \cdot (1 + \frac{\eta}{\textsc{Off}})\ ,\ \textsc{Opt} \cdot \log n\})$.

Beyond MTS: online matching on the line

Definition by picture: *n* servers; *n* online requests



Remarks

- This problem is not known to be an MTS
- Binding decisions: cannot pay to modify the matching
- Can *simulate* another matching

Analoguous prediction setup

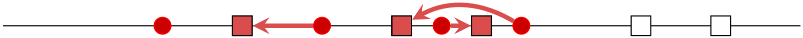
- Predict the set of severs matched in some $\textsc{Off}$
- Error: min cost of a perfect matching between these servers

Theorem

$\textsc{RobustFtP}$*'s analoguous costs* $O(\min\{\textsc{Off} \cdot (1 + \frac{\eta}{\textsc{Off}})\ ,\ \textsc{Opt} \cdot \log n\})$.

Beyond MTS: online matching on the line

Definition by picture: n servers; n online requests



Remarks

- This problem is not known to be an MTS
- Binding decisions: cannot pay to modify the matching
- Can *simulate* another matching

Analoguous prediction setup

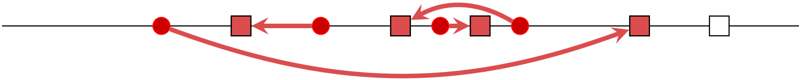
- Predict the set of severs matched in some $\textsc{Off}$
- Error: min cost of a perfect matching between these servers

Theorem

$\textsc{RobustFtP}$*'s analoguous costs* $O(\min\{\textsc{Off}\cdot(1+\frac{\eta}{\textsc{Off}})\ ,\ \textsc{Opt}\cdot\log n\})$.

Beyond MTS: online matching on the line

Definition by picture: n servers; n online requests



Remarks

- This problem is not known to be an MTS
- Binding decisions: cannot pay to modify the matching
- Can *simulate* another matching

Analoguous prediction setup

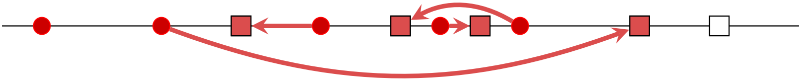
- Predict the set of severs matched in some $\textsc{Off}$
- Error: min cost of a perfect matching between these servers

Theorem

$\textsc{RobustFtP}$*'s analoguous costs* $O(\min\{\textsc{Off}\cdot(1+\frac{\eta}{\textsc{Off}})\ ,\ \textsc{Opt}\cdot\log n\})$.

Beyond MTS: online matching on the line

Definition by picture: n servers; n online requests



Remarks

- This problem is not known to be an MTS
- Binding decisions: cannot pay to modify the matching
- Can *simulate* another matching

Analoguous prediction setup

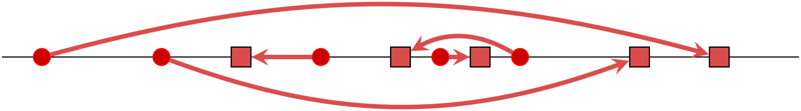
- Predict the set of severs matched in some $\textsc{Off}$
- Error: min cost of a perfect matching between these servers

Theorem

$\textsc{RobustFtP}$*'s analoguous costs* $O(\min\{\textsc{Off}\cdot(1+\frac{\eta}{\textsc{Off}})\ ,\ \textsc{Opt}\cdot\log n\})$.

Beyond MTS: online matching on the line

Definition by picture: *n* servers; *n* online requests



Remarks

- This problem is not known to be an MTS
- Binding decisions: cannot pay to modify the matching
- Can *simulate* another matching

Analoguous prediction setup

- Predict the set of severs matched in some $\textsc{Off}$
- Error: min cost of a perfect matching between these servers

Theorem

$\textsc{RobustFtP}$*'s analoguous costs* $O(\min\{\textsc{Off} \cdot (1 + \frac{\eta}{\textsc{Off}})\ ,\ \textsc{Opt} \cdot \log n\})$.

Conclusion

is useful !

MTS (and beyond)

- General prediction setup
- "Optimal" algorithm: RobustFtP

Caching

- Better algorithm in our setup: Trust&Doubt
- VS specific setup: similar guarantees & better on simulations
- With simple statistic predictors: better than LRU

Perspectives

- Develop better algorithms for other MTS
  (e.g., weighted caching, convex body chasing, *k*-server)