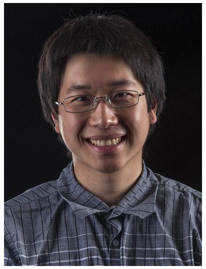
Memory Bounds for the Experts Problem

Vaidehi Srinivas
Northwestern University

Ziyu (Neil) Xu
Carnegie Mellon University

David P. Woodruff
Carnegie Mellon University

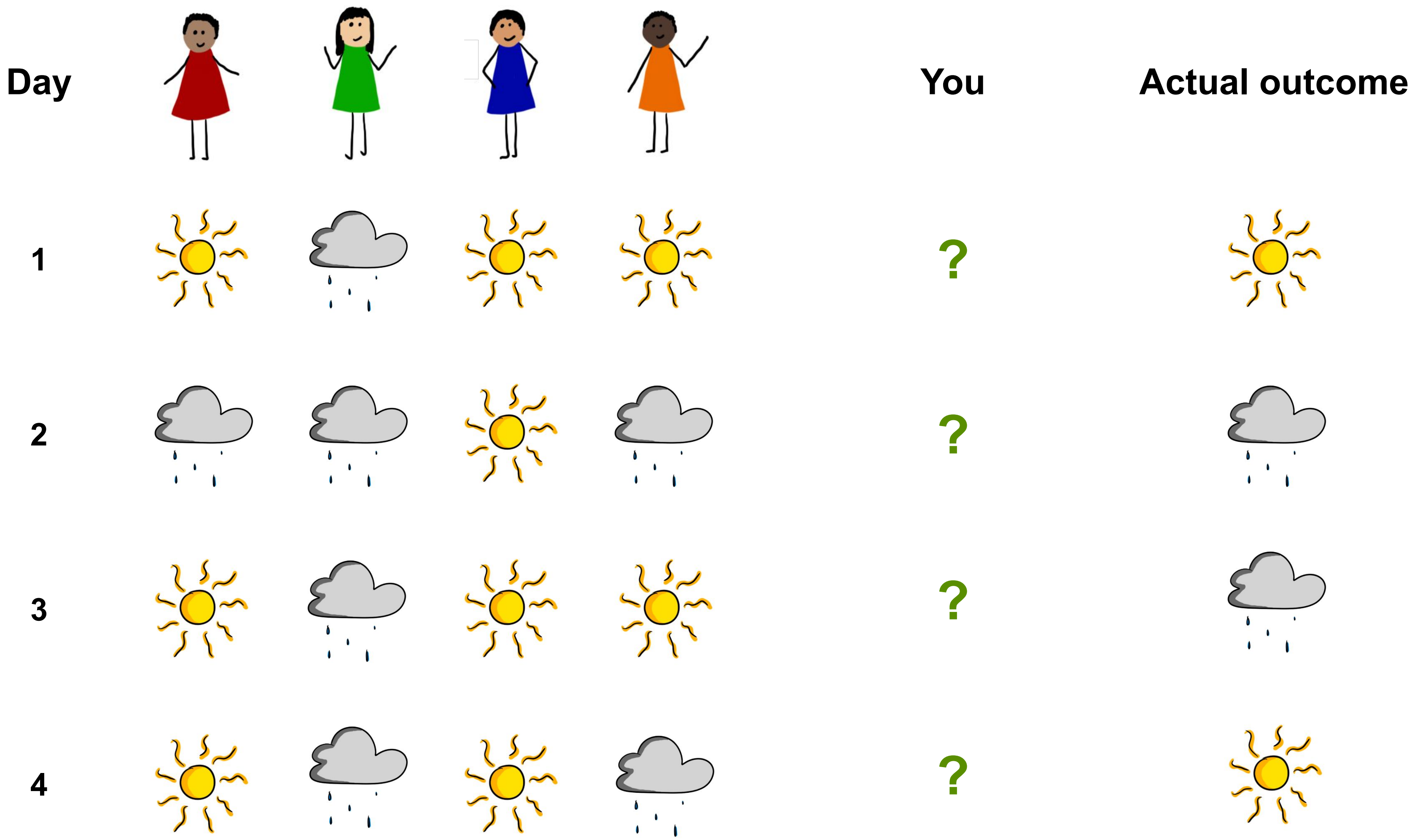
Samson Zhou
Carnegie Mellon University



arxiv.org/abs/2204.09837

Prediction with Expert Advice

a problem of **sequential prediction**



What does it mean to do well?

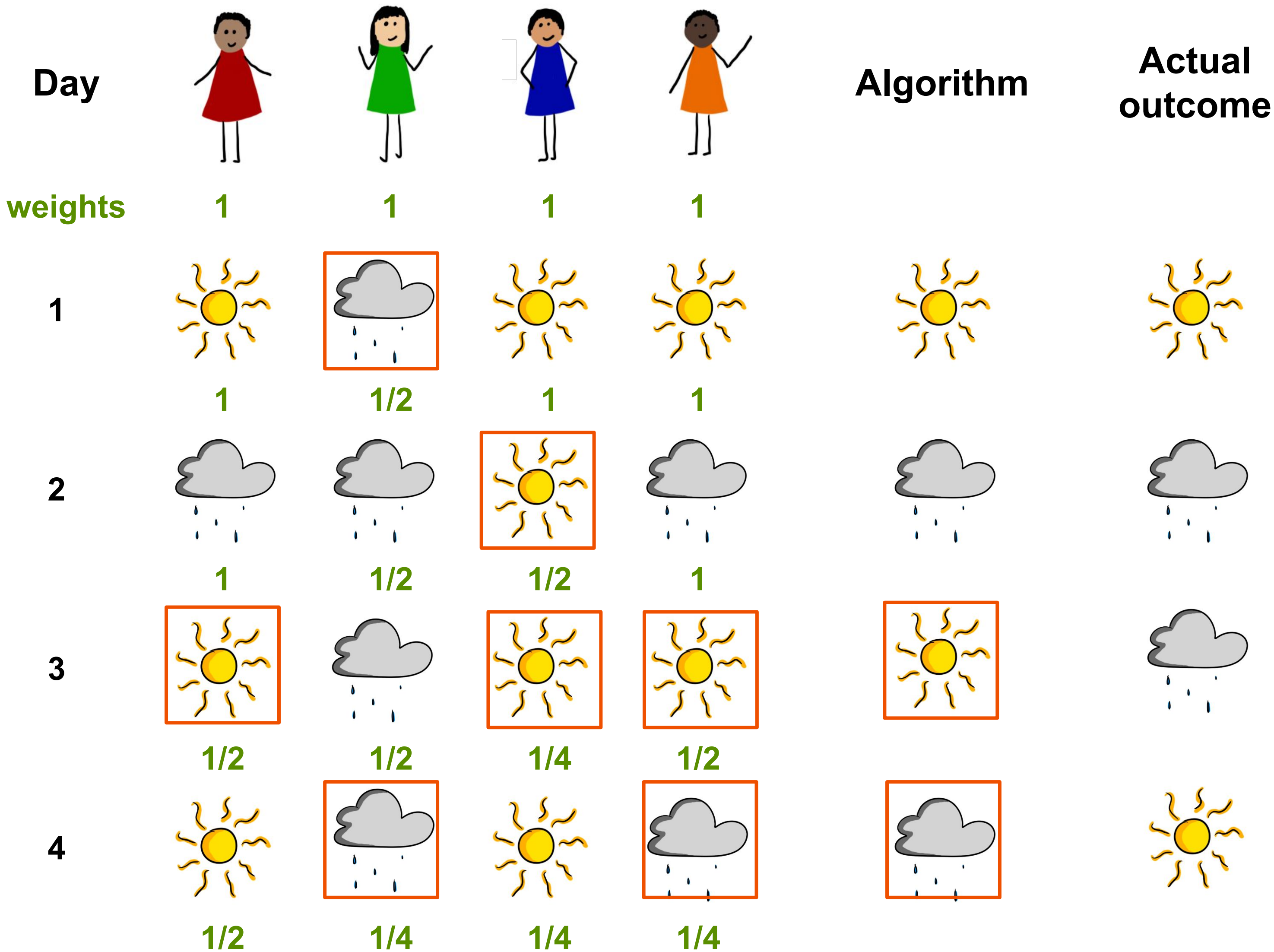
In general, predicting the future is impossible.

We judge our algorithm based on **regret**.

Definition (Regret)

$$\frac{\text{\# of mistakes algorithm makes more than the best expert}}{\text{\# of days}}$$

A Classical Algorithm: Multiplicative Weights



Algorithm makes 2 mistakes

Best expert makes 1 mistake

Regret: 1/4

Standard Guarantees for Multiplicative Weights

Theorem [Littlestone, Warmuth, '89] (Deterministic Weighted Majority)

# of mistakes by deterministic weighted majority $\leq 2.41\left(M + \log_2 n\right)$

regret: $O\left(\frac{M + \log n}{T}\right)$

where M is the # of mistakes the best expert makes, n is # of experts.

Theorem [Arora, Hazan, Kale, '12] (Standard Randomized MW)

For $\varepsilon > 0$, can construct algorithm A such that

expected # of mistakes by A $\leq (1 + \varepsilon) M + \frac{\ln n}{\varepsilon}$

regret: $O\left(\sqrt{\frac{\log n}{T}}\right)$

Information theoretically tight

Applications of the Experts Problem

- Ensemble learning, AdaBoost

- Online convex optimization

- Forecasting and portfolio optimization

- Solving for equilibria of zero sum games

Room for Improvement

Multiplicative weights keeps track of # of mistakes so far for every expert:

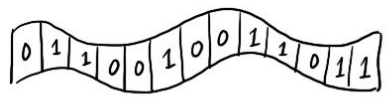
$\boldsymbol{\Omega}(n)$ memory

Can we solve this problem with less memory?

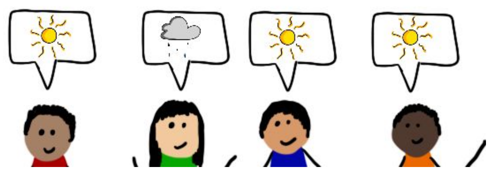
(we are willing to compromise on regret)

The Streaming Model

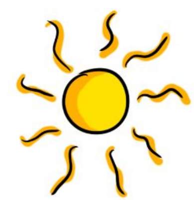
wake up with no memory



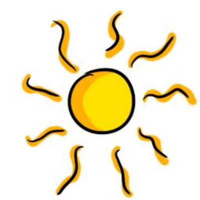
except a note from your past self
(at most ***s* bits**)



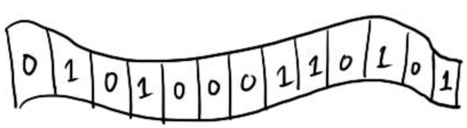
see expert predictions for today



make a prediction



see outcome



write a note to your future self
(at most ***s* bits**)

fall asleep and forget everything

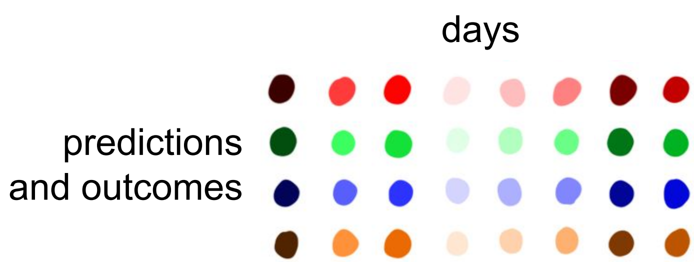
The Streaming Model

The complete sequence of T days is the **data stream**.

$$(\text{prediction}_1, \text{outcome}_1), \ldots, (\text{prediction}_T, \text{outcome}_T)$$

Definition (Arbitrary Order Model)

An adversary chose the predictions and outcomes to trick you.



Definition (Random Order Model)

An adversary chose the predictions and outcomes to trick you,

then the order of days was randomly shuffled.

Our Results

Theorem (Lower Bound)

Any algorithm that achieves average regret $\boldsymbol{\delta}$, in expectation, <u>must</u> use

$$\Omega\left(\frac{n}{\boldsymbol{\delta}^2 T}\right) \text{ space,}$$

even for random order and i.i.d. streams.

- To match Multiplicative Weights regret $\approx \sqrt{\frac{1}{T}}$, must use $\Omega(n)$ space
- Could potentially do much better for constant $\boldsymbol{\delta}$

Our Results

Theorem (Upper Bound for Random Order Streams)

For target $\boldsymbol{\delta} > \sqrt{\frac{16 \log^2 n}{T}}$, our algorithm achieves average regret $\boldsymbol{\delta}$ in expectation, using

$$\tilde{O}\left(\frac{n}{\boldsymbol{\delta}^2 T}\right) \text{ space.}$$

- Matches lower bound!

\

- Can handle general [0, 1] costs

Our Results

Theorem (Upper Bound for Arbitrary Order Streams)

For target $\boldsymbol{\delta} > \sqrt{\frac{128 \log^2 n}{T}}$, our algorithm achieves average regret $\boldsymbol{\delta}$, in expectation, using

$$\tilde{O}\left(\frac{n}{\boldsymbol{\delta} T}\right) \text{ space,}$$

when $M \leq \frac{\boldsymbol{\delta}^2 T}{1280 \log^2 n}$.

- Beats lower bound!
- Hardness of lower bound comes from regime where best expert makes large number of mistakes

Outline

1. Background and results

2. **Lower bound**

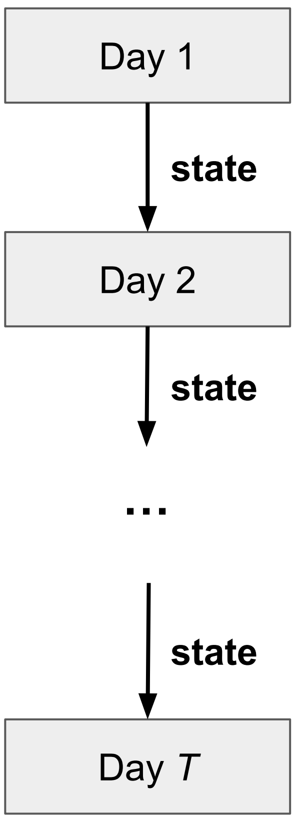
3. Upper bound

4. Conclusions

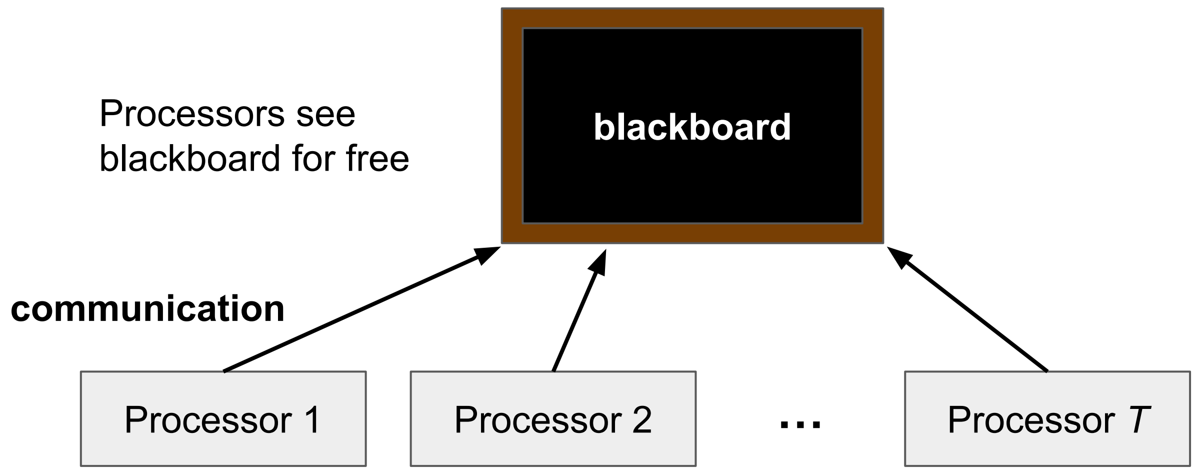
(Lower Bound)

Blackboard Communication Model

Streaming Model



Blackboard Model



S space streaming protocol $\Rightarrow$ $S \cdot T$ communication blackboard protocol

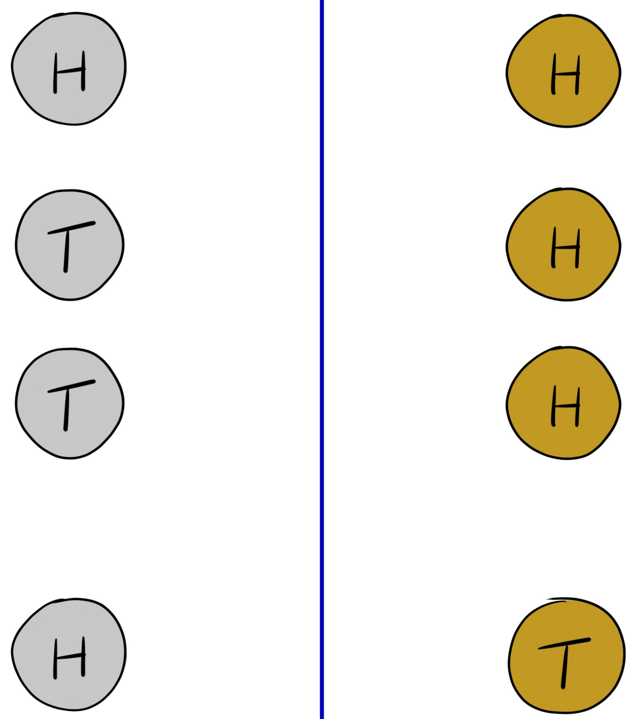
B blackboard lower bound $\Rightarrow$ B/T streaming lower bound

(Lower Bound)

The **δ**-Distributed Detection Problem

Distinguish between:

	FAIR COIN (pr. of heads ½)	BETTER COIN (pr. of heads ½ + **δ**)
Processor 1	H	H
Processor 2	T	H
Processor 3	T	H
…		
Processor T	H	T



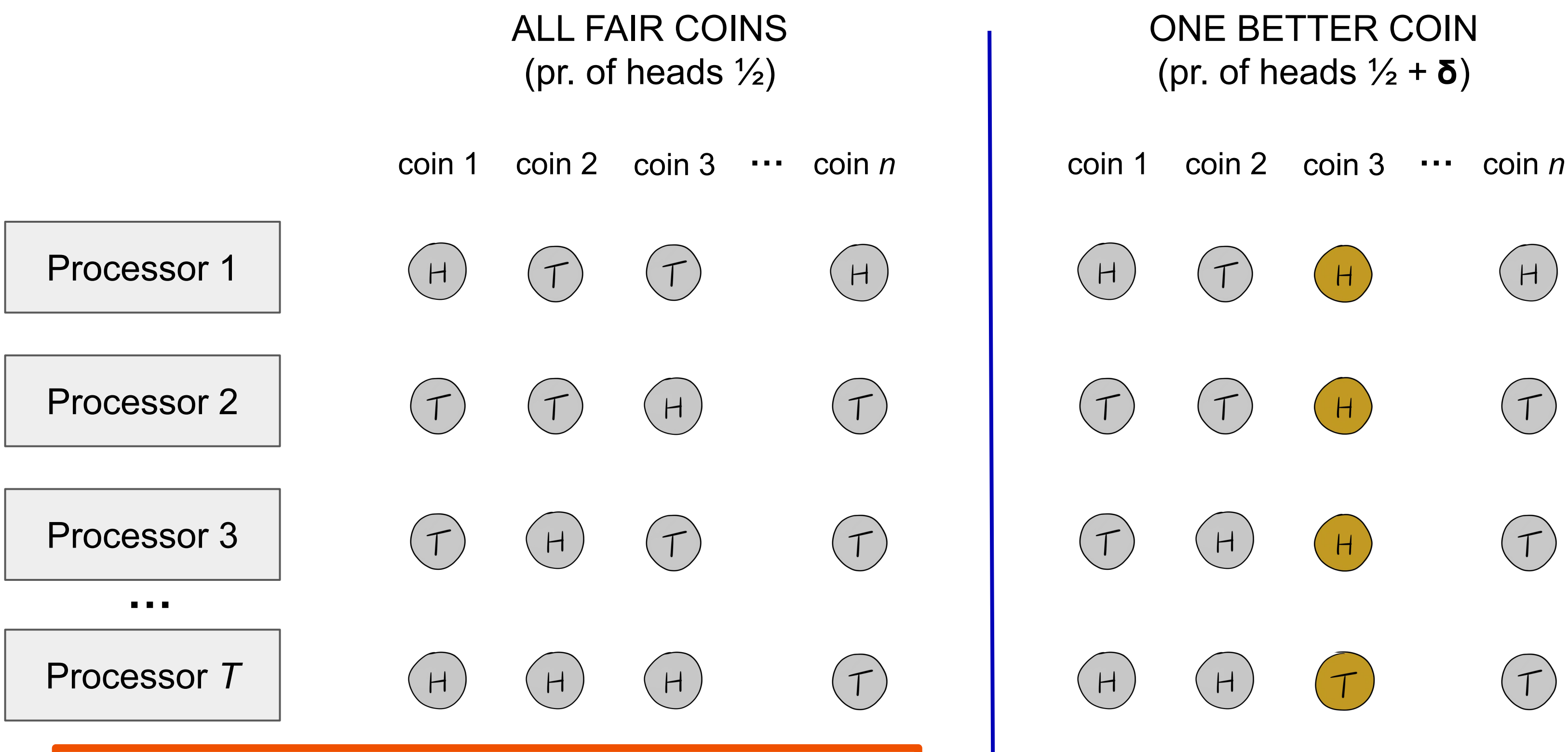
[Braverman, Garg, Ma, Nguyen, Woodruff, '16]

Requires
$\Omega(1/\boldsymbol{\delta}^2)$
communication in blackboard model

(Lower Bound)

The **δ**-DiffDist Problem

Distinguish between:



Requires
$\Omega(n/\delta^2)$
communication in blackboard model
(by Direct Sum Framework
[Bar-Yossef, Jayram, Kumar, Sivakumar, '04])

Can solve this using a δ-regret expert prediction algorithm!

(Lower Bound)

Reduction

"experts"

ALL FAIR COINS
(pr. of heads ½)

"days"	coin 1	coin 2	coin 3	…	coin n	"outcome"
Processor 1	H	T	T		H	H
Processor 2	T	T	H		T	H
Processor 3	T	H	T		T	H
…						
Processor T	H	H	H		T	H

ONE BETTER COIN
(pr. of heads ½ + **δ**)

"days"	coin 1	coin 2	coin 3	…	coin n	"outcome"
Processor 1	H	T	H		H	H
Processor 2	T	T	H		T	H
Processor 3	T	H	H		T	H
…						
Processor T	H	H	T		T	H

Best expert makes mistakes ½ of the time

Best expert makes mistakes ½ - δ of the time

WANT: **δ**-regret algorithm makes mistakes at least half the time

δ-regret algorithm makes mistakes less than half the time!

(Lower Bound)

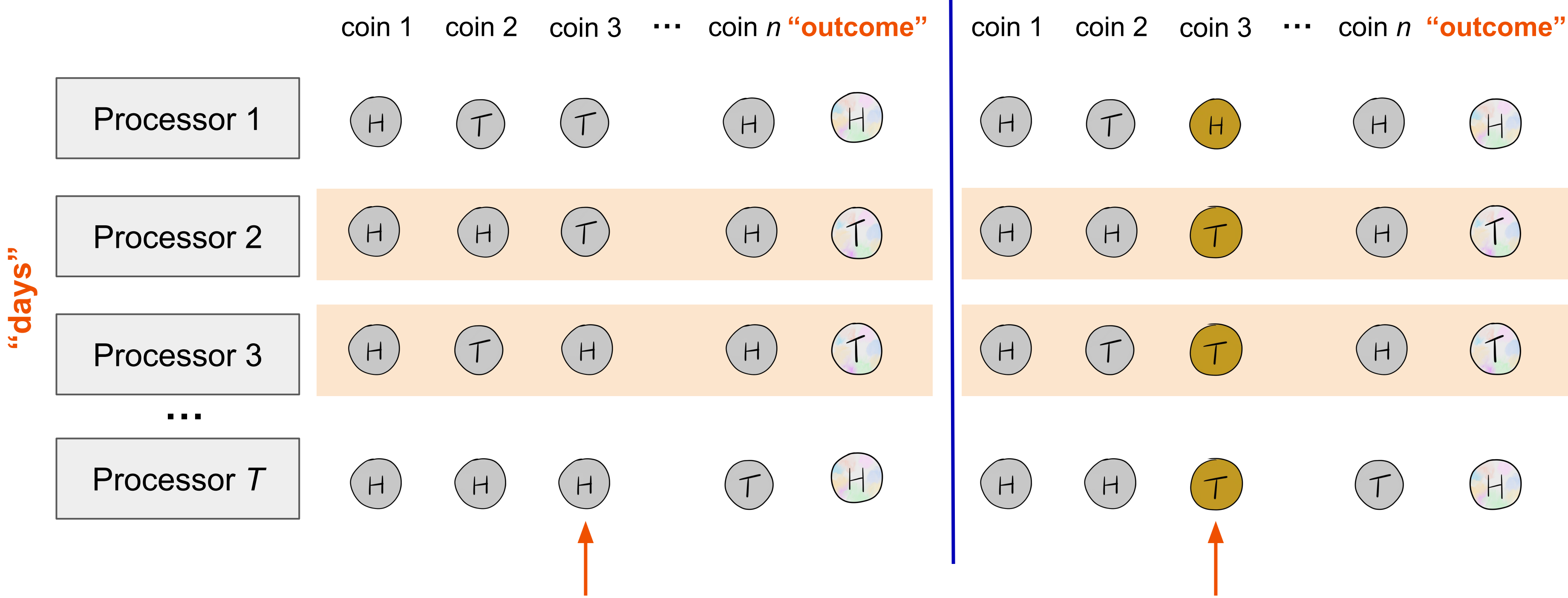
Reduction

Fix: on each day, **"mask"** predictions and outcomes with fair coin flip

"experts"

ALL FAIR COINS
(pr. of heads ½)

ONE BETTER COIN
(pr. of heads ½ + **δ**)



Best expert **still** makes
mistakes ½ of the time

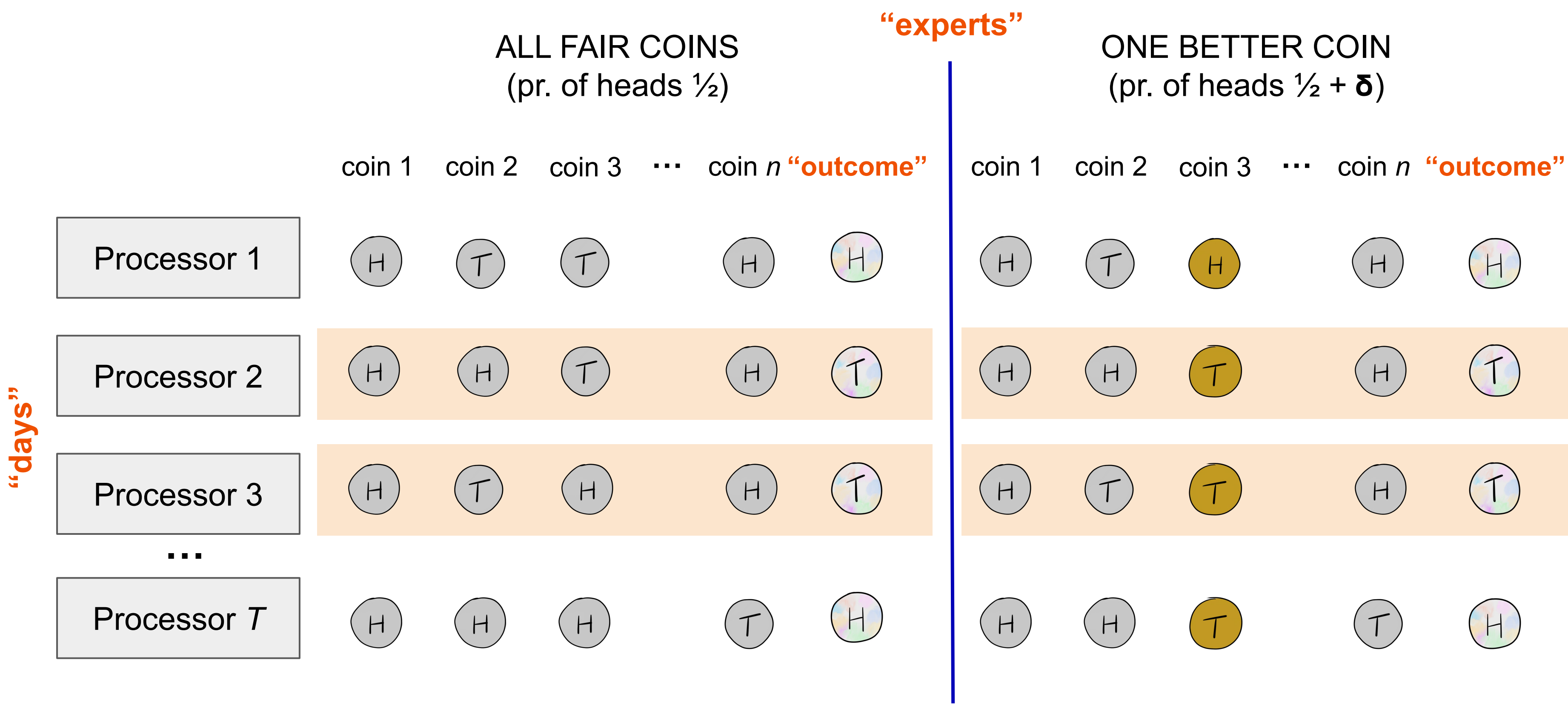
Best expert **still** makes
mistakes ½ - **δ** of the time

Outcome is uniform and independent of predictions,
no algorithm can make mistakes less than half the time!

δ-regret algorithm makes mistakes
less than half the time!

(Lower Bound)

Reduction



s : size of streaming state of **δ**-regret algorithm

This protocol for **δ**-DiffDist uses
$T\,(s + 1)$ bits of communication $\in \Omega\,(n/\delta^2)$

$$\boxed{s \in \Omega\left(\frac{n}{\delta^2 T}\right)}$$

Outline

1. Background and results
2. Lower bound
3. **Upper bound**
4. Conclusions

(Upper Bound)

Sampling-based Algorithm

Recall (Multiplicative Weights):

cost of best expert

$$\text{cost} \leqslant \underbrace{(1+\boldsymbol{\delta})\,M}_{\text{"ongoing" cost}} + \underbrace{\frac{\ln n}{\boldsymbol{\delta}}}_{\text{"start-up" cost}}$$

Goal: Given guess for M, make at most $M + O(\boldsymbol{\delta} T)$ mistakes

Strategy:

- Run MW on a randomly sampled "pool" of $\frac{n \ln n}{\boldsymbol{\delta}^2 T}$ experts
- If/when every expert in pool has average mistake rate $\geqslant \frac{M}{T} + \boldsymbol{\delta}$, resample and start over

Total Cost:

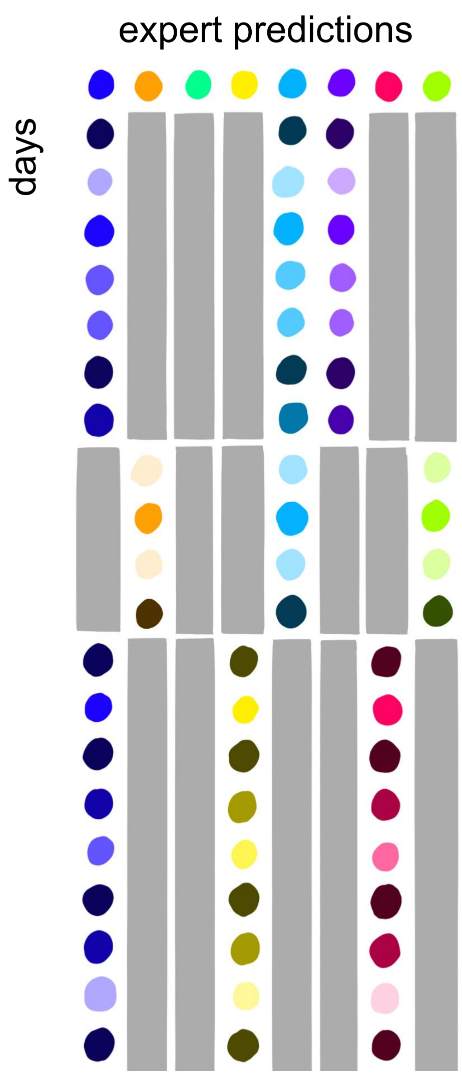
$$\underbrace{(1+\boldsymbol{\delta})\left[\,M + \boldsymbol{\delta} T\,\right]}_{M + O(\boldsymbol{\delta} T)} + \frac{\ln n}{\boldsymbol{\delta}} \underbrace{\big(\#\text{ of rounds of sampling}\big)}_{\text{just need to bound this!}}$$

(Upper Bound)

Algorithm Sketch

Idea: run MW on randomly sampled subset of experts

set **threshold** of “acceptable” error rate



+ **start-up cost**

best expert in set **doing well**

MW does well

everyone in set doing badly: **resample!**

+ **start-up cost**

best expert in set **doing well**

MW does well

everyone in set doing badly: **resample!**

+ **start-up cost**

best expert in set **doing well**

MW does well

Analysis: we won’t restart too many times

(Upper Bound)

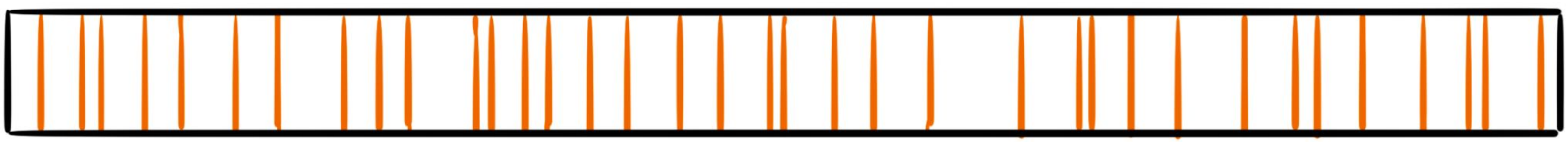
Bounding # of Rounds

Recall: Total Cost

$$M + O(\boldsymbol{\delta} T) \;+\; \frac{\ln n}{\boldsymbol{\delta}}\,\big(\#\text{ of rounds}\big)$$

Random Order Streams: with high probability, once we catch the best expert, we never resample.

Mistakes of best expert over time



evenly spaced

Probability we catch the best expert in given pool:

$$\frac{\text{size of pool}}{n} = \frac{n \ln n \,/\, (\boldsymbol{\delta}^2 T)}{n}$$

Expected # of rounds:

$$\frac{\boldsymbol{\delta}^2 T}{\ln n}$$

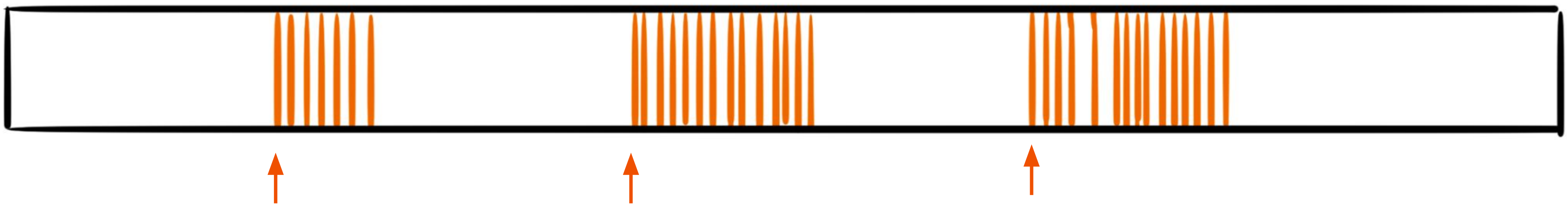
Total cost: $M + O(\boldsymbol{\delta} T)$

(Upper Bound)

Bounding # of Rounds

Arbitrary Order Streams: Assume best expert makes $O(\frac{\boldsymbol{\delta}^2 T}{\log^2 n})$ mistakes

Mistakes of best expert over time



bad days to sample

Few mistakes means few bad days.

Even if we sample on every bad day, we will be okay!

Outline

1. Background and results

2. Lower bound

3. Upper bound

4. **Conclusions**

Our Results (recap)

Theorem (Lower Bound)

Any algorithm that achieves average regret $\boldsymbol{\delta}$, in expectation, <u>must</u> use

$$\Omega\left(\frac{n}{\boldsymbol{\delta}^2 T}\right) \text{ space,}$$

even for random order and i.i.d. streams.

Theorem (Upper Bound for Random Order Streams)

For target $\boldsymbol{\delta} > \sqrt{\frac{16 \log^2 n}{T}}$, our algorithm achieves average regret $\boldsymbol{\delta}$ in expectation, using

$$\tilde{O}\left(\frac{n}{\boldsymbol{\delta}^2 T}\right) \text{ space.}$$

Theorem (Upper Bound for Arbitrary Order Streams)

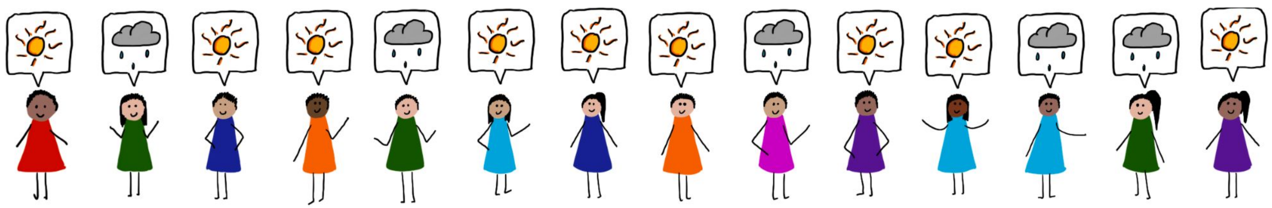
For target $\boldsymbol{\delta} > \sqrt{\frac{128 \log^2 n}{T}}$, our algorithm achieves average regret $\boldsymbol{\delta}$, in expectation, using

$$\tilde{O}\left(\frac{n}{\boldsymbol{\delta} T}\right) \text{ space,}$$

when $M \leq \frac{\boldsymbol{\delta}^2 \mathrm{T}}{1280 \log^2 n}$.

Future Directions

- Tight bounds for arbitrary-order streams
  - For constant regret **δ**, can we tolerate best expert making constant fraction of mistakes?
- Better bounds when expert costs have more structure
  - i.e. expert predictions are real numbers that are evaluated against true outcome with some loss function
- Better bounds with extra constraints on experts



Thanks!