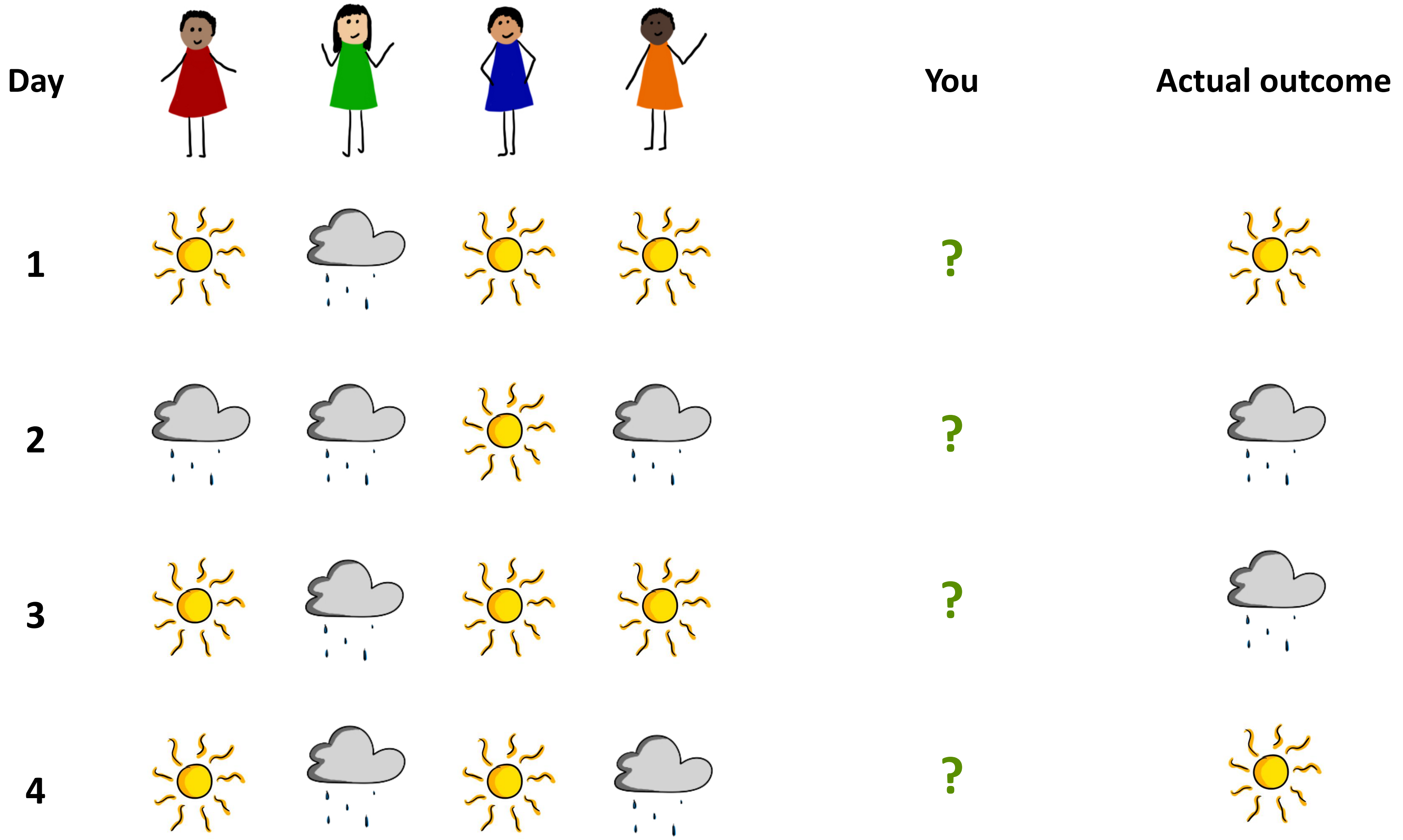
Memory Bounds for the Expert Problem

Vaidehi Srinivas
David P. Woodruff
Ziyu (Neil) Xu
Samson Zhou

Prediction with Expert Advice

a fundamental problem of **sequential prediction**



Quantifying Performance

Make no distributional assumptions

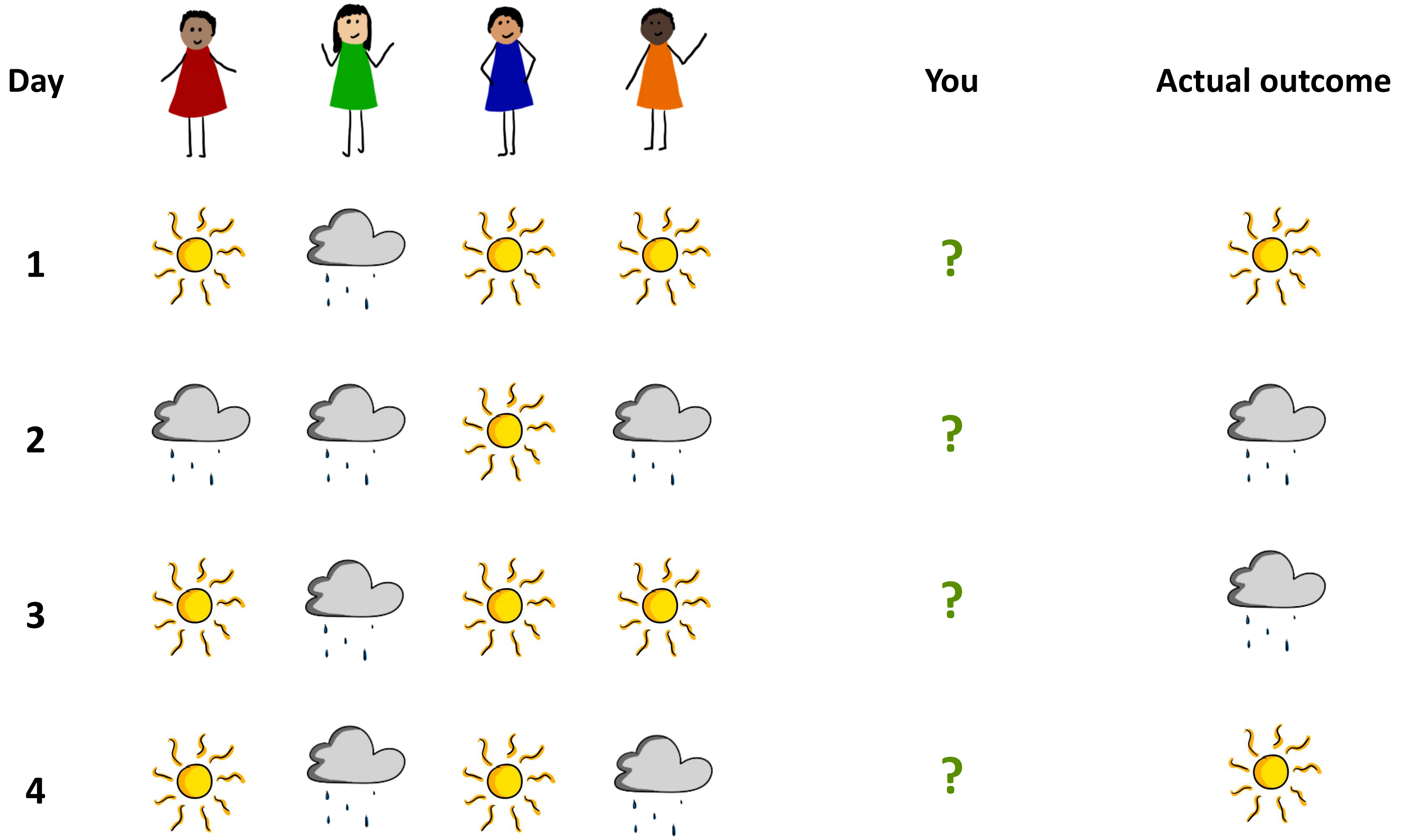
We judge our algorithm based on **regret**.

Definition (Regret)

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

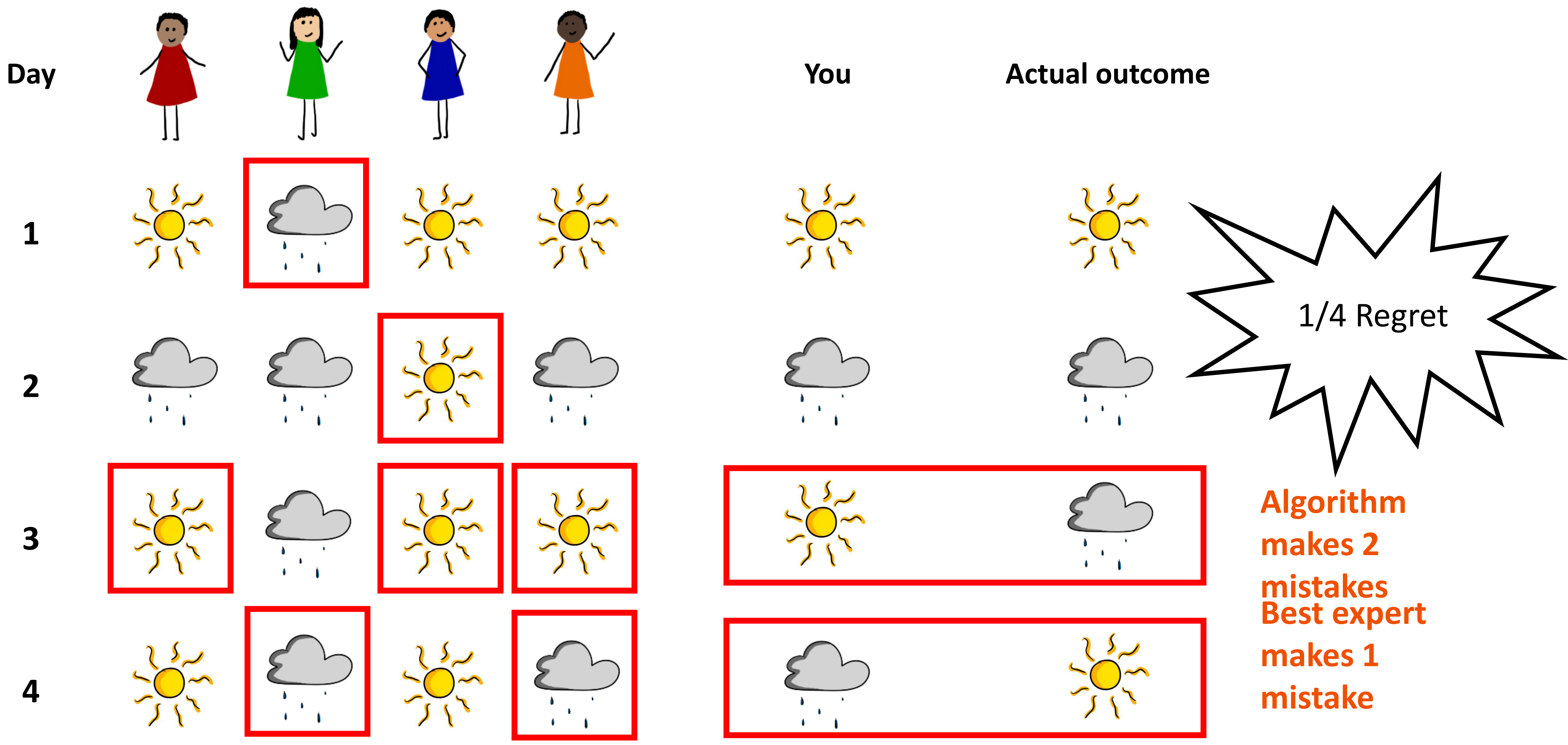
Prediction with Expert Advice

a fundamental problem of **sequential prediction**



Prediction with Expert Advice

a fundamental problem of **sequential prediction**



The Online Learning with Experts Problem

- n experts who decide either $\{0,1\}$ on each of T days ($n \gg T$)
- Algorithm takes advice from experts and predict either $\{0,1\}$ on each day
- Algorithm sees the outcome, which is either $\{0,1\}$, of each day and can use this information on future days
- The cost of the algorithm is the number of incorrect predictions

- Regret is (# of mistakes we make - M)/T, i.e., the amortized additional cost of the algorithm compared to the cost M of the best expert

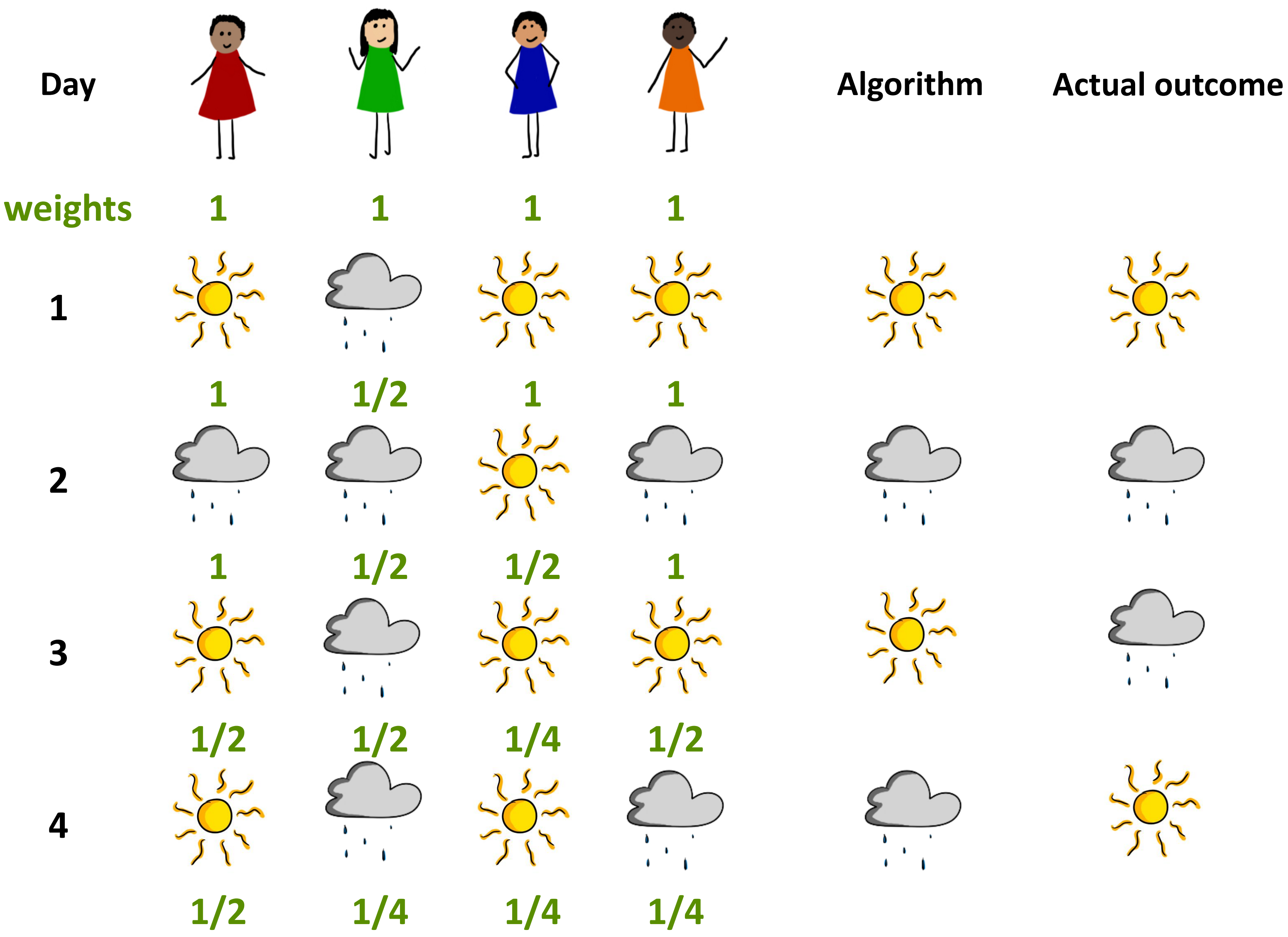
Applications of the Experts Problem

- Ensemble learning, e.g., AdaBoost

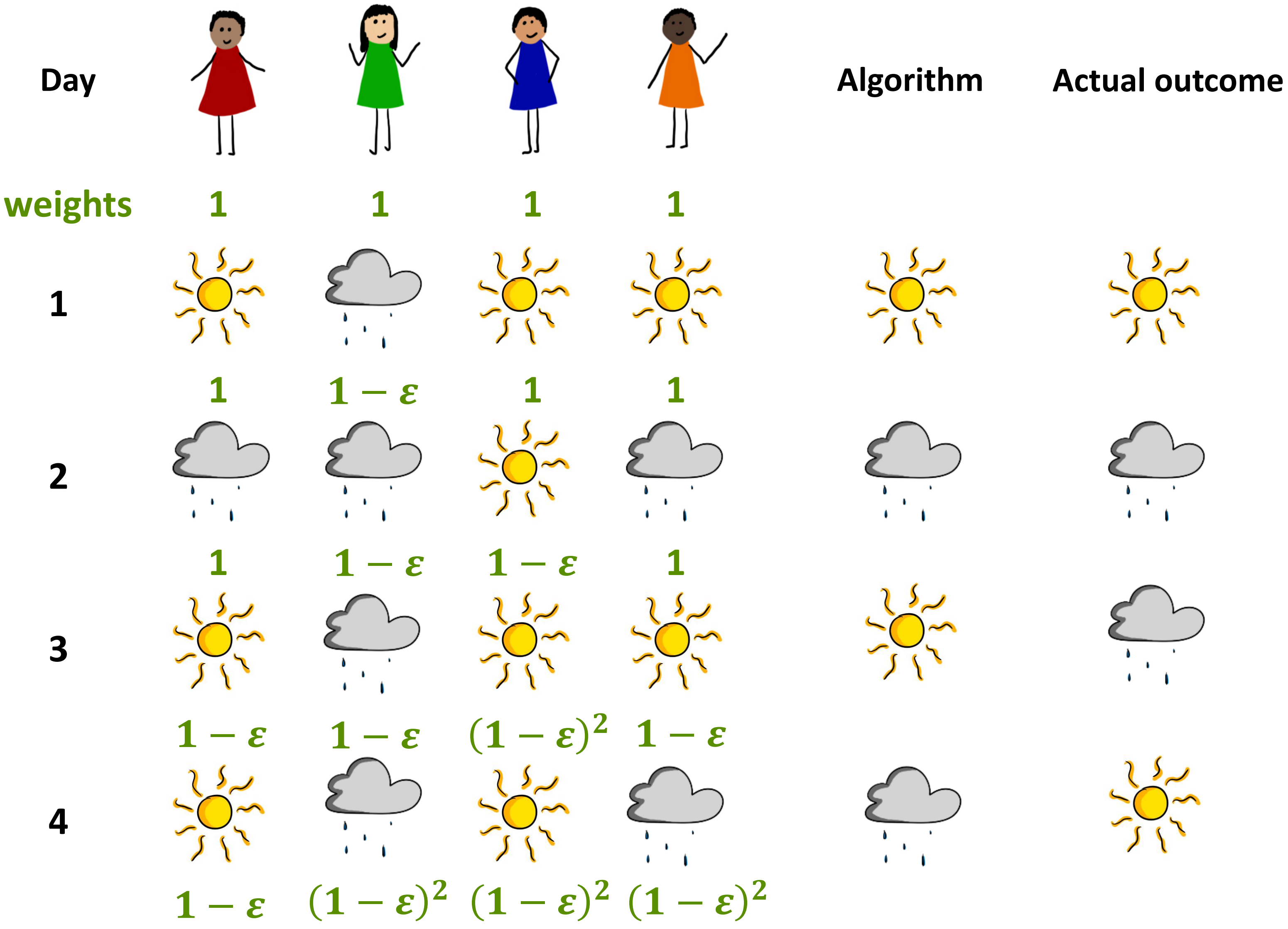
- Forecast and portfolio optimization

- Special case of online convex optimization

Weighted Majority (Littlestone, Warmuth 89)



Weighted Majority (Littlestone, Warmuth 89)



Guarantee for Weighted Majority

Theorem (Deterministic Weighted Majority)

# of mistakes by deterministic weighted majority $\leq \quad (2+\varepsilon)M + \frac{2}{\varepsilon} \ln \ n$

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

- $(1-\varepsilon)^M \leq$ sum of the weights $\leq \left(1-\frac{\varepsilon}{2}\right)^m n$

Guarantee for Weighted Majority

Theorem (Deterministic Weighted Majority)

# of mistakes by deterministic weighted majority $\leq \quad (2+\varepsilon)\mathrm{M} + \frac{2}{\varepsilon} \ln\ n$

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

Theorem (Randomized Weighted Majority, i.e, Multiplicative Weights)

For ε > 0, can construct algorithm A such that

$$\text{E[\# of mistakes by } A\text{]} \leq (1+\varepsilon) M + \frac{O(\ln n)}{\varepsilon}$$

Previous Work

- Weighted majority algorithm down-weights each expert that is incorrect on each day and selects the weighted majority as the output
- Weighted majority algorithm gets $(2+\varepsilon)M + \frac{O(\log n)}{\varepsilon}$ total mistakes
- Randomized weighted majority algorithm randomly follows each expert with probability proportional to the weight of the expert
- Randomized weighted majority algorithm achieves regret $O\left(\sqrt{\frac{\log n}{T}}\right)$

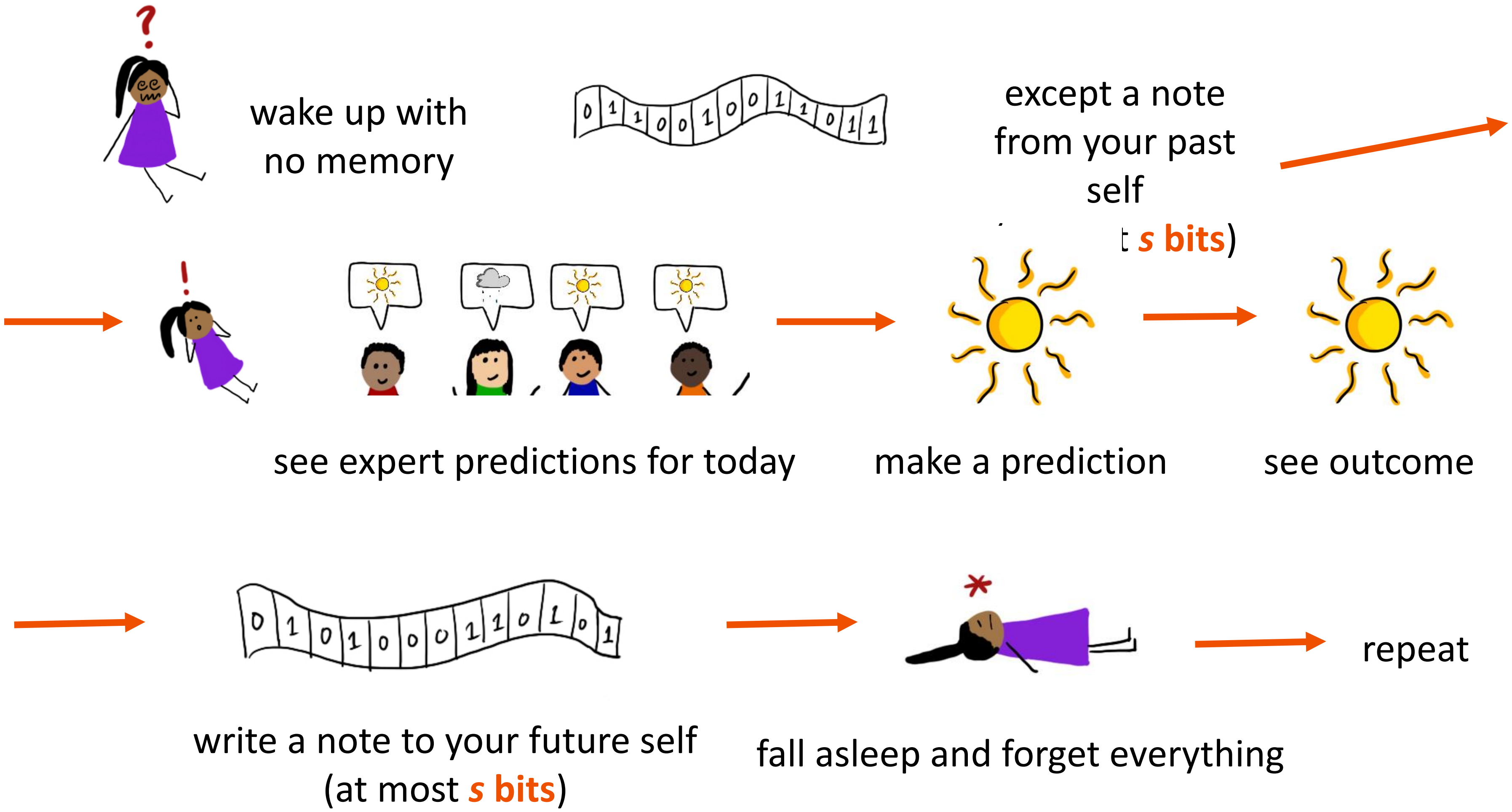
Memory Bounds for the Expert Problem

- These algorithms require $\Omega(n)$ memory to maintain weights for each expert – but what if n is very large and we want sublinear space?

- Can use no memory and just randomly guess each day – still good if the best expert makes a lot of mistakes but bad if the best expert makes very few mistakes

- What are the space/accuracy tradeoffs for the online learning with experts problem?

The Streaming Model



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 chooses a worst-case ordering of the days and outcomes in the stream beforehand.

Definition (Random Order Model)

An adversary chooses worst-case ordering of the outcomes, <u>then</u> the order of days is randomly shuffled.

A Natural Idea

- What if we just identify the best expert?

- Find the best expert so far, follow it until a new best expert emerges, identify the new best expert, find it, repeat
  - Doesn't even work in offline setting

- Could do weighted majority, but uses $\Omega(n)$ space

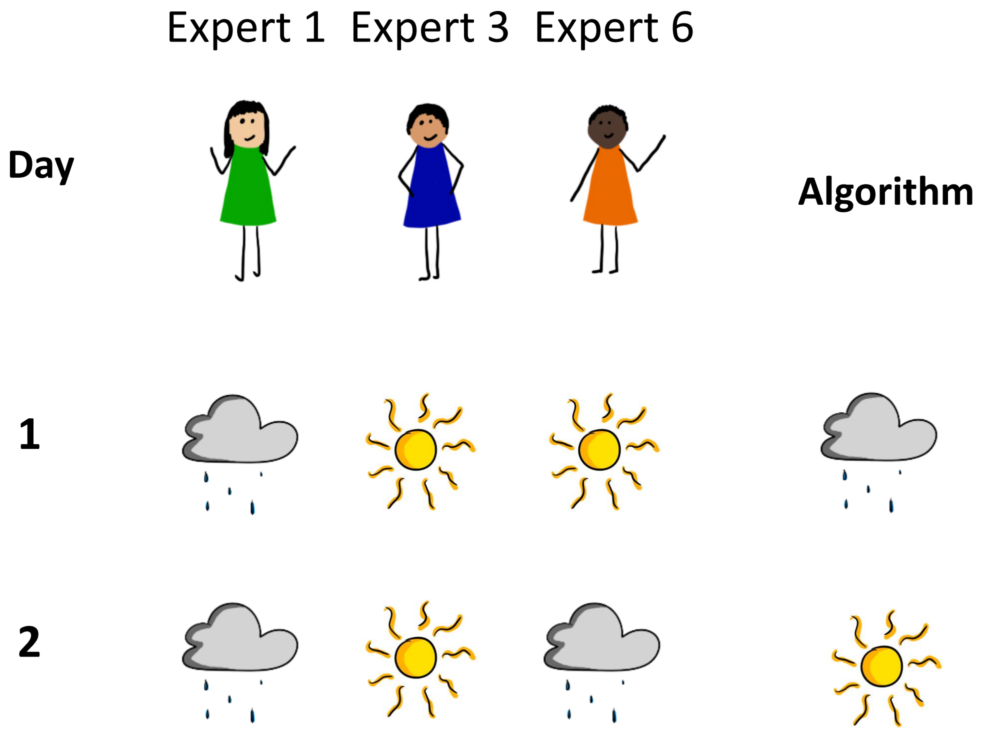
Set Disjointness Communication Problem

- Set disjointness communication problem: Alice has a set $X \in \{0,1\}^n$ and Bob has a set $Y \in \{0,1\}^n$ and the promise is that either $|X \cap Y| = 0$ or $|X \cap Y| = 1$

- Set disjointness requires total (randomized) communication $\Omega(n)$

Reduction

- Holds even for 2 days (can copy each day T/2 times if desired)
- Alice creates a stream S so that each element of X is an expert that is correct on day 1
- Bob creates a stream S' so that each element of Y is an expert that is correct on day 2



Reduction

- Alice runs streaming algorithm A on the stream S created by their set X and passes the state of A to Bob, who continues running the algorithm on the stream S' created by their set Y

- At the end, A will output an expert $i \in [n]$, and then Alice and Bob will check whether $X \cap Y = i$

- Solves set disjointness* so A must use $\Omega(n)$ space

- Not end of story: low-regret algorithm need not find best expert, even if second best expert makes half as many mistakes

Our Results (I)

- Any algorithm that achieves $\delta < \frac{1}{2}$ regret with probability at least $\frac{3}{4}$ must use $\Omega\left(\frac{n}{\delta^2 T}\right)$ space

- Lower bound holds for arbitrary-order, random-order, and i.i.d. streams

Our Results (II)

- There exists an algorithm that uses $O\left(\frac{n}{\delta^2 T}\log^2 n \log\frac{1}{\delta}\right)$ space and achieves expected regret $\delta > \sqrt{\frac{8\log n}{T}}$ in the random-order model

- The algorithm is almost-tight with the space lower bounds and oblivious to M, the number of mistakes made by the best-expert

- Can achieve regret almost matching randomized weighted majority

- Result extends to general costs in $[0, \rho]$ with expected regret $\rho\delta$

Our Results (III)

- For $M < \frac{\delta^2 T}{1280 \log^2 n}$ and $\delta > \sqrt{\frac{128 \log^2 n}{T}}$, there exists an algorithm that uses $\tilde{O}\left(\frac{n}{\delta T}\right)$ space and achieves regret δ with probability $\frac{4}{5}$

- The algorithm *beats* the lower bounds, showing that the hardness comes from the best expert making a "lot" of mistakes
- Can achieve regret almost matching randomized weighted majority
- The algorithm oblivious to M, the number of mistakes made by the best expert

Format

- Part 1: Background
- Part 2: Lower Bound
- Part 3: Arbitrary Model
- Part 4: Random-Order Model

Questions?

Lower Bound

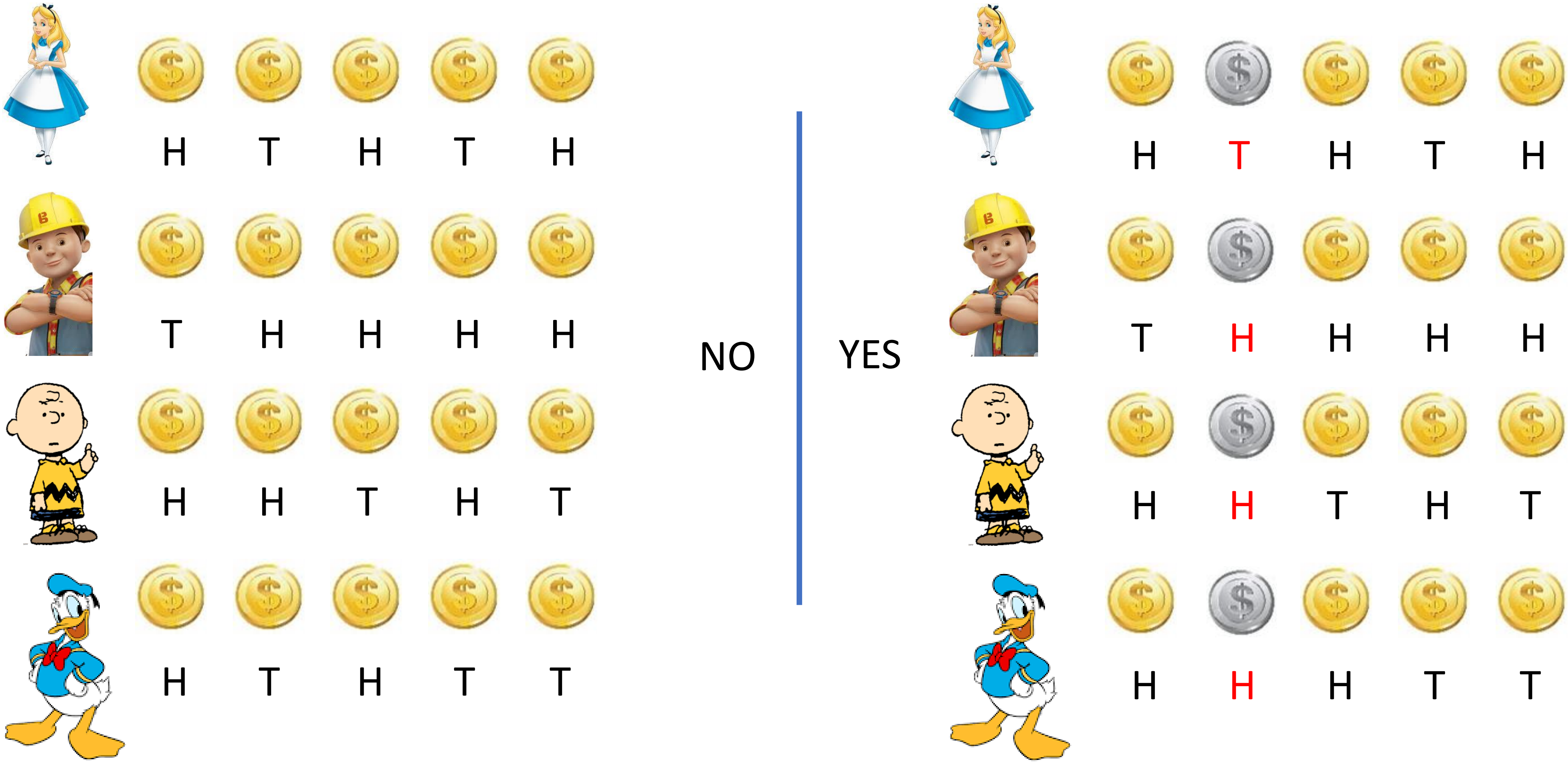
- Any algorithm that achieves $\delta < \frac{1}{2}$ (average) regret with probability at least $\frac{3}{4}$ must use $\Omega\left(\frac{n}{\delta^2 T}\right)$ space

- Lower bound holds for arbitrary-order, random-order, and i.i.d. streams

Communication Problem for Lower Bound

- Distributed detection problem
- ε-DIFFDIST problem: T players each hold n bits and must distinguish between two cases.
- Case 1: (NO) Every index for every player is drawn i.i.d. from a fair coin, i.e., a Bernoulli distribution with parameter $\frac{1}{2}$
- Case 2: (YES) An index $L \in [n]$ is selected arbitrarily. The L-th bit of each player is chosen i.i.d. from a Bernoulli distribution with parameter $\frac{1}{2} + \varepsilon$ and all the other bits are chosen i.i.d. from a fair coin

Communication Problem for Lower Bound

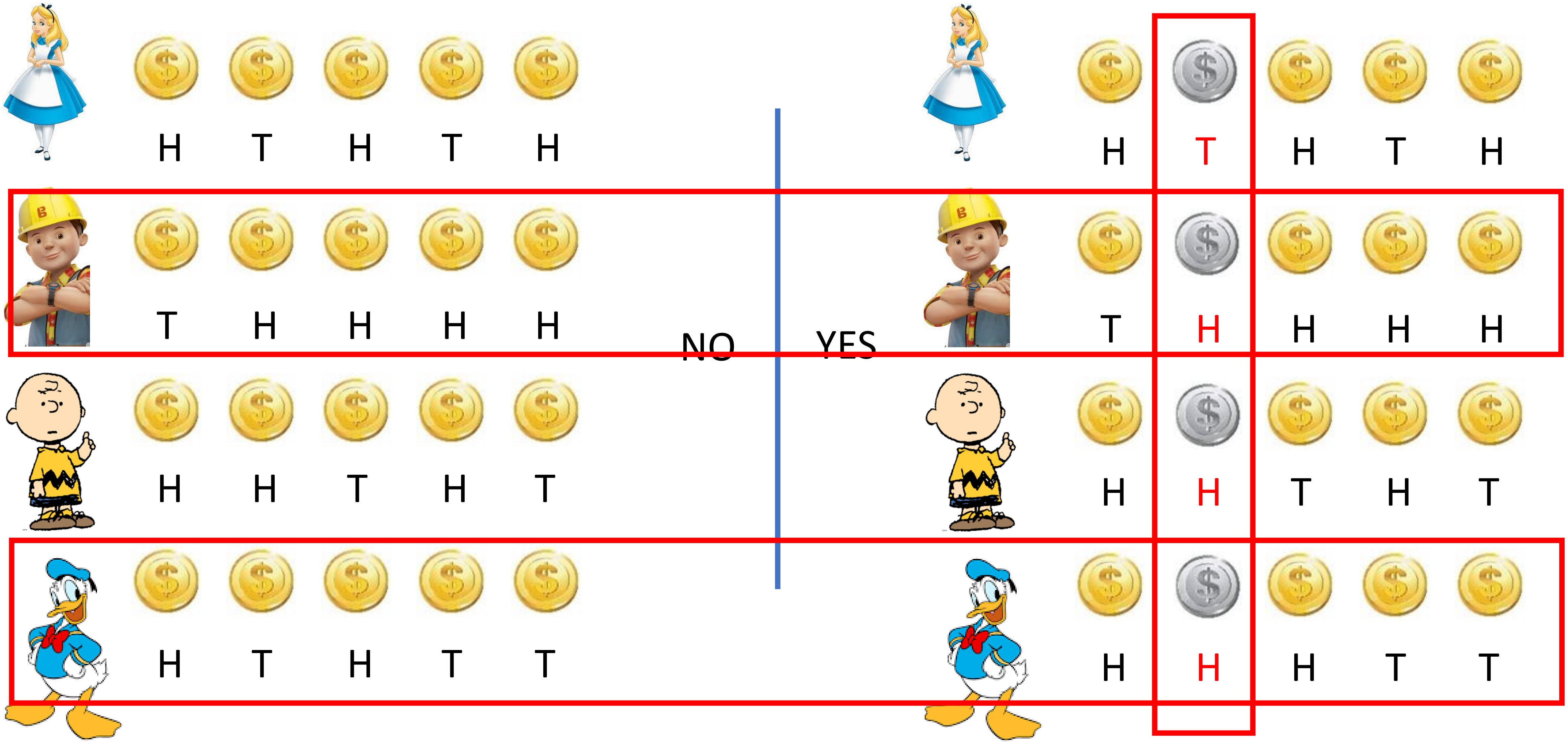


ε-DIFFDIST Problem

- ε-DIFFDIST problem: T players each hold n bits and must distinguish between two cases.

- Protocol: Randomly choose $\tilde{O}\left(\frac{1}{\varepsilon^2}\right)$ players and send all bits of those players, see whether some bit has bias at least $\frac{\varepsilon}{2}$

Communication Problem for Lower Bound



ε-DIFFDIST Problem

- ε-DIFFDIST problem: T players each hold n bits and must distinguish between two cases.
- Protocol: Randomly choose $\tilde{O}\left(\frac{1}{\varepsilon^2}\right)$ players and send all bits of those players, see whether some bit has bias at least $\frac{\varepsilon}{2}$
- Communication of protocol: $\tilde{O}\left(\frac{n}{\varepsilon^2}\right)$
- Theorem: $\Omega\left(\frac{n}{\varepsilon^2}\right)$ communication is necessary

ε-DIFFDIST Problem

- Theorem: $\Omega\left(\frac{n}{\varepsilon^2}\right)$ communication is necessary

- Fact: $\Omega\left(\frac{1}{\varepsilon^2}\right)$ samples are necessary to distinguish between a fair coin, i.e., a Bernoulli distribution with parameter $\frac{1}{2}$ and a coin with bias ε

- Intuition: players sort of need to solve the single coin problem on each of the n coins (actually just need the OR)

ε-DIFFDIST Problem

- Formally, all the coins are independent in the NO distribution

- Can use a direct sum theorem for OR [BJKS04], so reduces to showing high information cost under NO distribution on a single coin

- $\Omega\left(\frac{1}{\varepsilon^2}\right)$ information necessary to distinguish between a single fair coin, i.e., a Bernoulli distribution with parameter $\frac{1}{2}$ and a coin with bias ε, even when information is measured under the NO distribution
  - Uses strong data processing inequality [ZDJW13, GMN14, BGM+16]

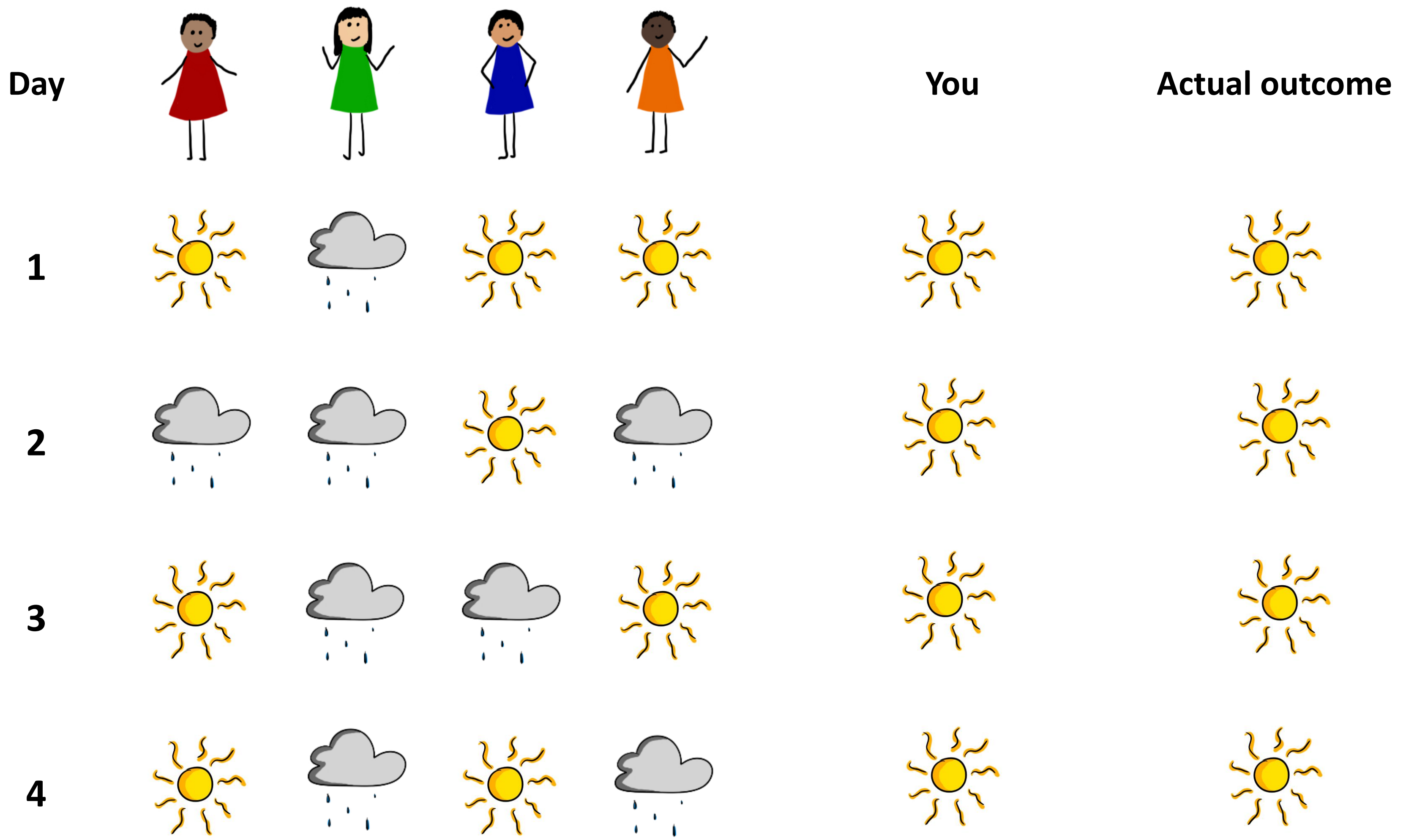
ε-DIFFDIST Summary

- ε-DIFFDIST problem: T players each hold n bits and must distinguish between two cases.
- Case 1: (NO) Every index for every player is drawn i.i.d. from a fair coin, i.e., a Bernoulli distribution with parameter $\frac{1}{2}$
- Case 2: (YES) An index $L \in [n]$ is selected arbitrarily. The L-th bit of each player is chosen i.i.d. from a Bernoulli distribution with parameter $\frac{1}{2} + \varepsilon$ and all the other bits are chosen i.i.d. from a fair coin
- Fact: $\Omega\left(\frac{n}{\varepsilon^2}\right)$ communication is necessary to solve the problem

Reduction Intuition

- Each player in the ε-DIFFDIST Problem corresponds to a different day
- Each bit in the ε-DIFFDIST Problem corresponds to a different expert
- Reduction: distinguishing whether there exists a slightly biased random bit corresponds to distinguishing whether there exists a slightly "better" expert

Reduction Challenge

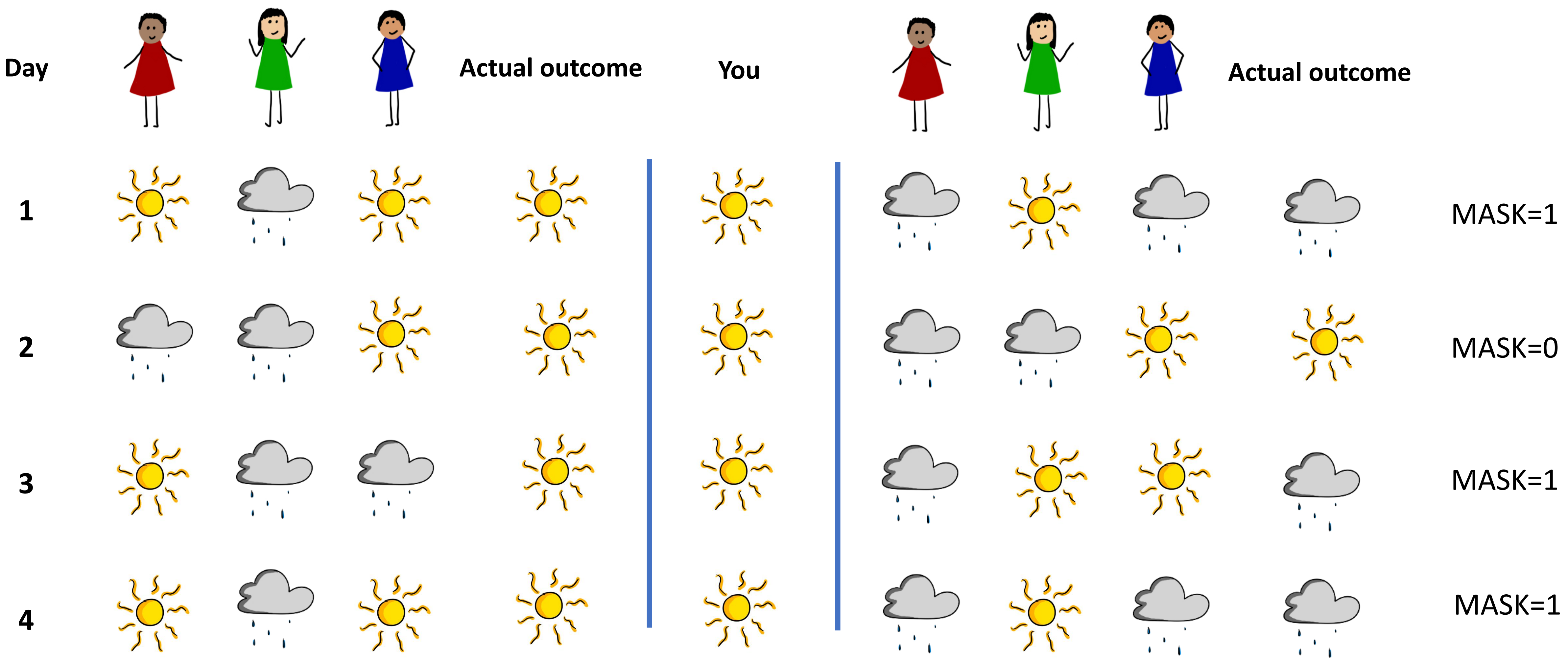


Reduction

- We would like to use an online learning with experts algorithm for solving ε-DIFFDIST Problem for $\varepsilon = O(\delta)$ by sampling $\Omega\left(\frac{1}{\delta^2}\right)$ players
- However, an algorithm with bad guarantees can still “luckily” have good cost

- Use masking argument – outcome of each day is masked by an independent fair coin flip on each day (expert advice also flipped)

Reduction Challenge



Reduction

- For constant $\delta < \frac{1}{2}$, if there is no biased coin, no expert will do better than $\frac{1}{2} + \frac{\delta}{3}$ with probability at least $\frac{1}{4}$
- For constant $\delta < \frac{1}{2}$, if there is a biased coin, an expert will do better than $\frac{1}{2} + \frac{2\delta}{3}$ with probability at least $\frac{1}{4}$

Reduction Summary

- The online learning with experts algorithm with regret δ will be able to solve the ε-DIFFDIST Problem with probability at least $\frac{3}{4}$ for $\varepsilon = O(\delta)$. Must use $\Omega\left(\frac{n}{\delta^2}\right)$ total communication

- Any algorithm that achieves $\delta < \frac{1}{2}$ regret with probability at least $\frac{3}{4}$ must use $\Omega\left(\frac{n}{\delta^2 T}\right)$ space

Format

- Part 1: Background
- Part 2: Lower Bound
- Part 3: Arbitrary Model
- Part 4: Random-Order Model

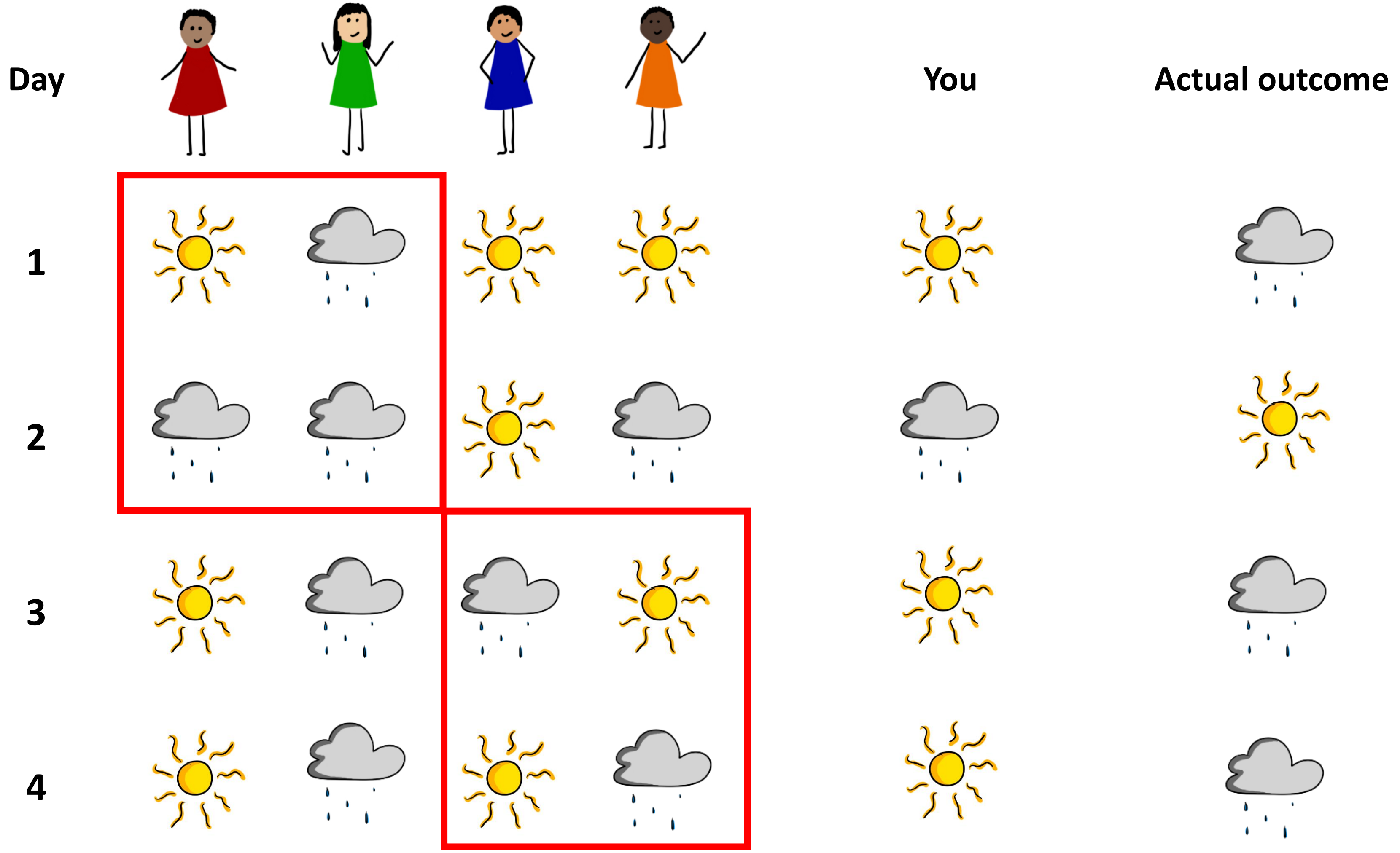
Questions?

No Mistake Regime

- For $M < \frac{\delta^2 T}{1280 \log^2 n}$ and $\delta > \sqrt{\frac{128 \log^2 n}{T}}$, there exists an algorithm that uses $\tilde{O}\left(\frac{n}{\delta T}\right)$ space and achieves regret δ with probability $\frac{4}{5}$

- We know there is a really accurate expert. What if we iteratively pick "pools" of experts and delete them if they run "poorly"?

Reduction Problem



No Mistake Regime

- If iteratively pick pool of next k experts (“rounds”) and output the majority vote of the pool while deleting any incorrect expert, each pool will have at most $O(\log k)$ errors

- If best expert makes no mistakes, use $\frac{n}{k}$ pools to achieve regret δT means setting $k = \tilde{O}\left(\frac{n}{\delta T}\right)$

No Mistake Regime Summary

- Algorithm: Iteratively pick pool of next $k = \tilde{O}\left(\frac{n}{\delta T}\right)$ experts ("rounds") and output the majority vote of the pool while deleting any incorrect expert

- If the number of rounds is small, the pools must have done well so the overall regret is small
- The number of rounds cannot be large because at some point the best expert would have been sampled and retained

“Low-Mistake” Regime

- Algorithm: Iteratively pick pool of next $k = \tilde{O}\left(\frac{n}{\delta T}\right)$ experts (“rounds”) and output the majority vote of the pool while deleting any incorrect expert

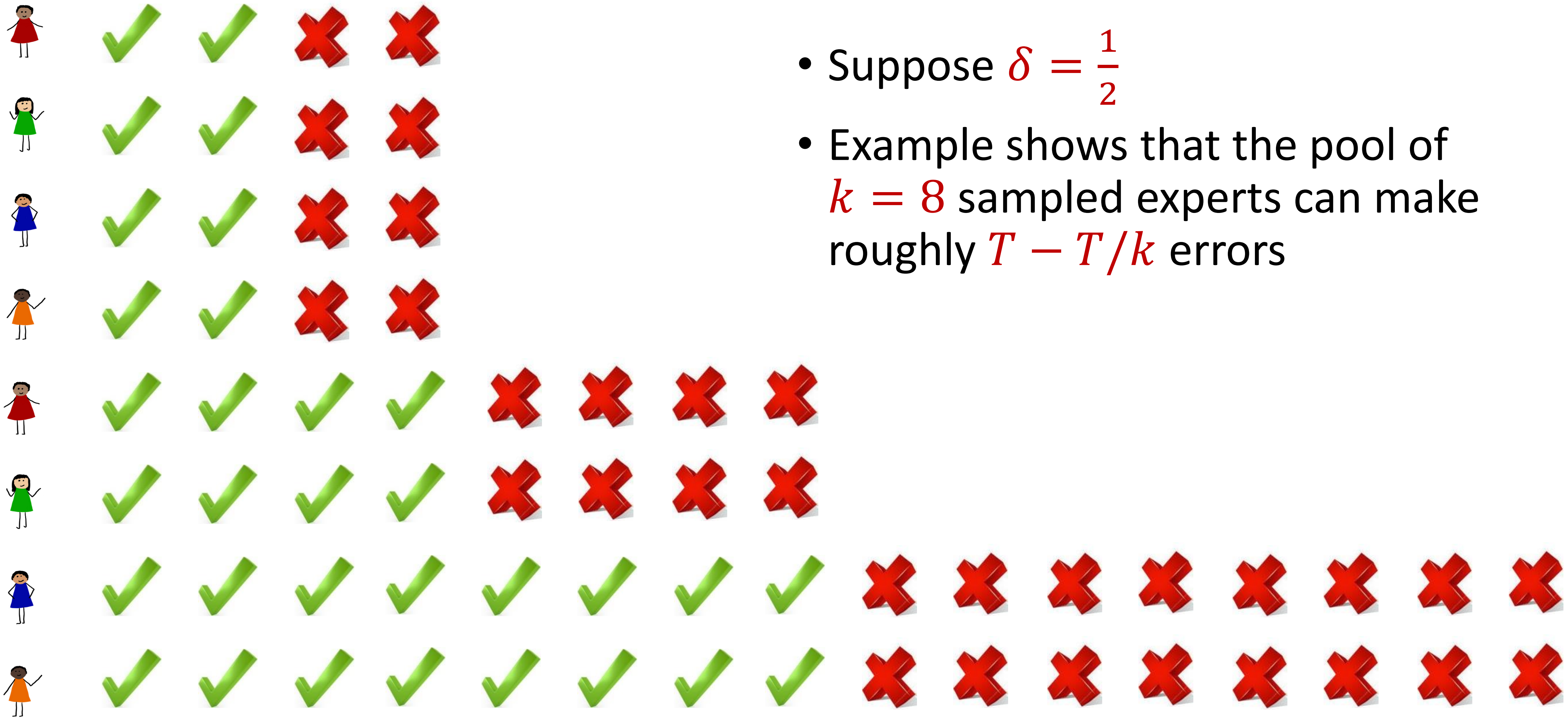
- If best expert makes M mistakes, use $\frac{nM}{k}$ pools to achieve regret δT means setting $k = \tilde{O}\left(\frac{nM}{\delta T}\right)$, but this is too large!

"Low-Mistake" Fix-Its

- Fix #1: Randomly sample pools of experts instead of iteratively picking pools
- Problem #1: Cannot guarantee that the best expert will be retained

- Fix #2: Delete experts that have erred with fraction at least $1 - \delta$
- Problem #2: "Build-up" of errors

A Really Bad Case Study



- Suppose $\delta = \frac{1}{2}$
- Example shows that the pool of $k = 8$ sampled experts can make roughly $T - T/k$ errors

"Low-Mistake" Regime

- Algorithm: Repeatedly sample a pool of $k = \tilde{O}\left(\frac{n}{\delta T}\right)$ experts and output the majority vote of the pool while deleting any expert with lower than $1 - \frac{\delta}{8 \log n}$ accuracy since it was sampled

WANT TO SHOW

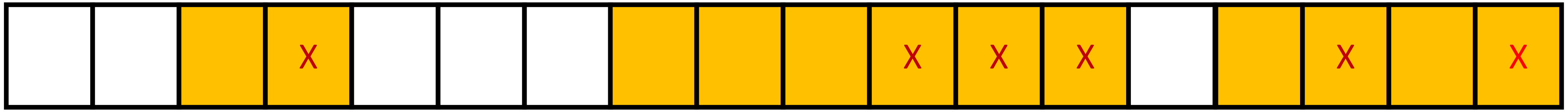
- If the number of rounds is small, the pools must have done well so the overall regret is small
- The number of rounds cannot be large because at some point the best expert would have been sampled and retained

“Low-Mistake” Regime: First Property

- Algorithm: Repeatedly sample a pool of $k = \tilde{O}\left(\frac{n}{\delta T}\right)$ experts and output the majority vote of the pool while deleting any expert with lower than $1 - \frac{\delta}{8 \log n}$ accuracy since it was sampled
- Lemma: For $\delta > \sqrt{\frac{128 \log^2 n}{T}}$, a pool that is used for t days can only make $\frac{t\delta}{2} + 4 \log n$ mistakes
- For the algorithm to make $T\delta$ mistakes, need at least $\frac{T\delta}{8 \log n}$ rounds

“Low-Mistake” Regime: Second Property

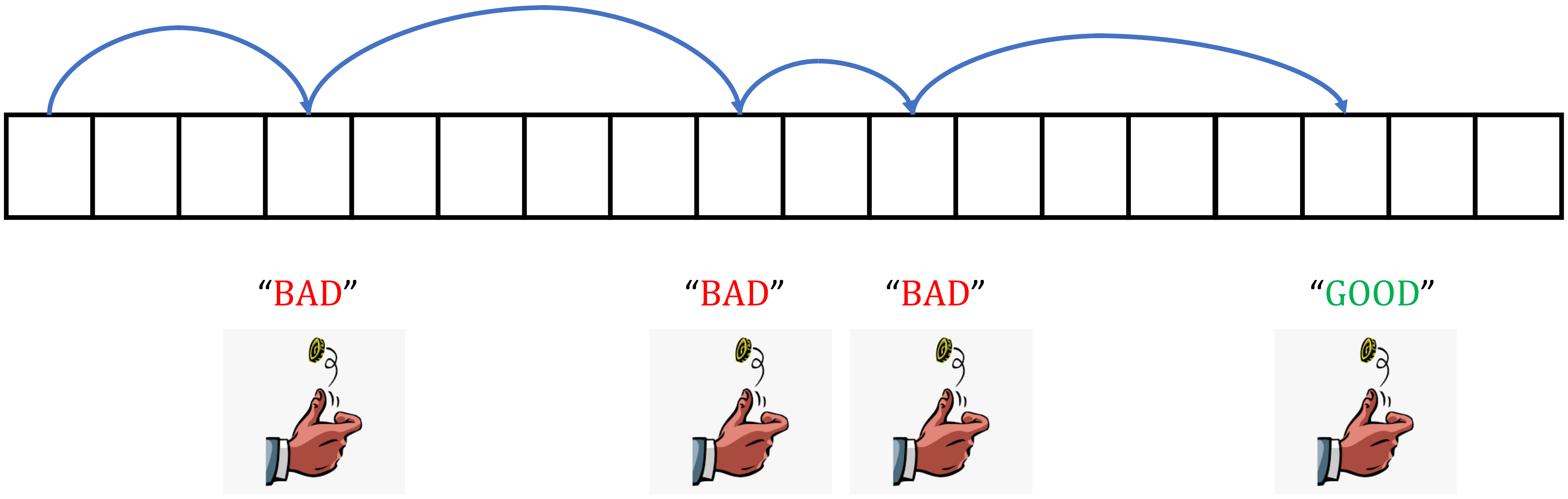
- For the algorithm to make $T\delta$ mistakes, need at least $\frac{T\delta}{8\log n}$ rounds
- “BAD” day: the best expert is deleted by the pool if it is sampled on that day



- $|\mathrm{BAD}| \leq \frac{8M\log n}{\delta}$

"Low-Mistake" Regime: Second Property

- For the algorithm to make $T\delta$ mistakes, need at least $\frac{T\delta}{8 \log n}$ rounds
- A bad algorithm must not sample the best expert on a "GOOD" day

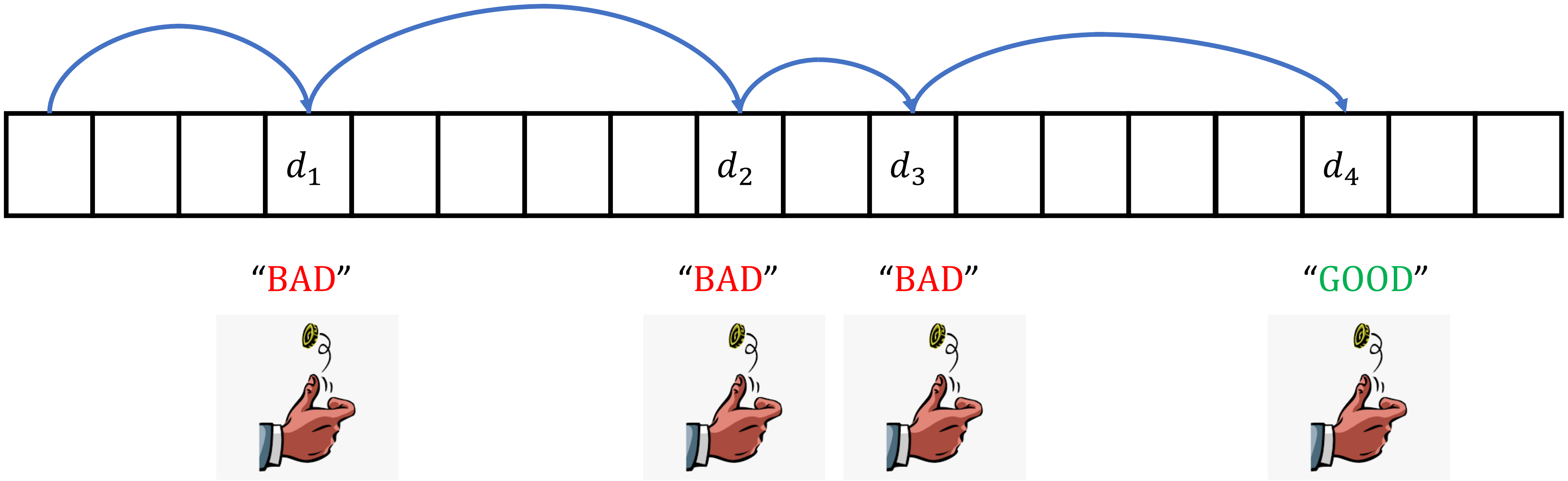


“Low-Mistake” Regime: Second Property

- For the algorithm to make $T\delta$ mistakes, need at least $\frac{T\delta}{8\log n}$ rounds
- Must avoid sampling the best expert on $\Omega\left(\frac{T\delta}{\log n}\right)$ rounds
- $O\left(\frac{n\log^2 n}{\delta T}\right)$ experts sampled in each round → low probability

Analysis

- Define a set of random variables $d_1, d_2, \ldots$ for each round's day
- Given d_i, draw d_{i+1} from the distribution of possible days for the next round based on possible experts sampled in the pool conditioned on entire history



Arbitrary Order Model Summary

- Algorithm: Repeatedly sample a pool of $k = \tilde{O}\left(\frac{n}{\delta T}\right)$ experts and output the majority vote of the pool while deleting any expert with lower than $1 - \frac{\delta}{8 \log n}$ accuracy since it was sampled

- If the number of rounds is small, the pools must have done well so the overall regret is small

- The number of rounds cannot be large because at some point the best expert would have been sampled and retained

Format

- Part 1: Background
- Part 2: Lower Bound
- Part 3: Arbitrary Model
- Part 4: Random-Order Model

Questions?

Random-Order Streams

- There exists an algorithm that uses $O\left(\frac{n}{\delta^2 T}\log^2 n \log\frac{1}{\delta}\right)$ space achieves expected regret $\delta > \sqrt{\frac{8\log n}{T}}$ in the random-order model

TAKING A STEP BACK

- We used majority vote of remaining experts in sampled pool
- Instead of removing experts, could just downweight them and run deterministic weighted majority
- Why not randomized weighted majority, i.e., multiplicative weights?

Multiplicative Weights Algorithm

Algorithm 4 The multiplicative weights algorithm.

Input: Number n of experts, number T of rounds, parameter ε

1: Initialize $w_i^{(1)} = 1$ for all $i \in [n]$.
2: **for** $t \in [T]$ **do**
3: $\quad p_i^{(t)} \leftarrow \frac{w_i^{(t)}}{\sum_{i \in [n]} w_i^{(t)}}$
4: $\quad$ Follow the advice of expert i with probability $p_i^{(t)}$.
5: $\quad$ Let $c_i^{(t)}$ be the cost for the decision of expert $i \in [n]$.
6: $\quad w_i^{(t+1)} \leftarrow w_i^{(t)} \left(1 - \varepsilon c_i^{(t)}\right)$
7: **end for**

- Theorem (Arora, Hazan, Kale 2012): Expected number of mistakes by the algorithm is at most $\frac{\ln n}{\varepsilon} + (1 + \varepsilon)M$

Random-Order Streams

- Algorithm: Repeatedly sample a pool of $k = \tilde{O}\left(\frac{n}{\delta^2 T}\right)$ experts and run multiplicative weights on pool, resample if the expected cost of the pool over t time "is bad"

- Can compute this expected cost, so if it doesn't follow the theory, it means you didn't sample the best expert

Random-Order Streams

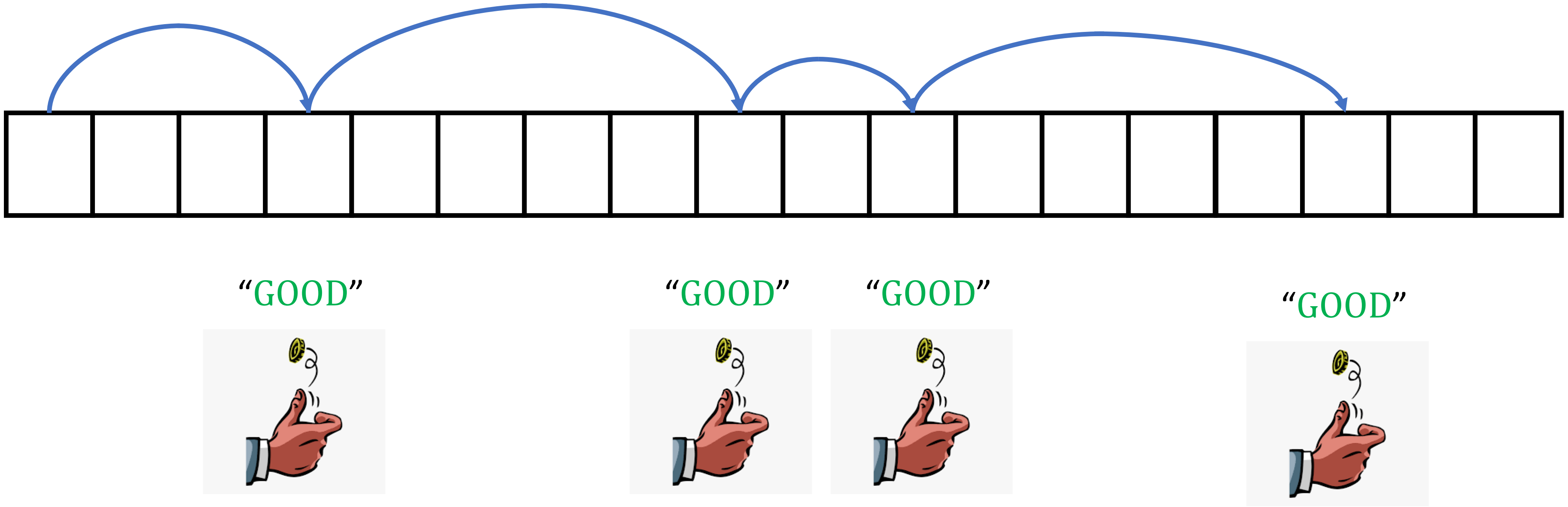
- Algorithm: Repeatedly sample a pool of $k = \tilde{O}\left(\frac{n}{\delta^2 T}\right)$ experts and run multiplicative weights on pool, resample if the expected cost of the pool over t time “is bad”.

WANT TO SHOW

- If the number of rounds is small, the pools must have done well so the overall regret is small
- The number of rounds cannot be large because at some point the best expert would have been sampled and retained

Random-Order Idea

- Enforce
  - (1) the algorithm will do well when the pool contains the best expert
  - (2) we will never delete the pool if it contains the best expert
- The number of rounds cannot be large because at some point the best expert would have been sampled and retained



Summary of Multiplicative Weights Algorithm

- There exists an algorithm that uses $O\left(\frac{n}{\delta^2 T}\log^2 n\right)$ space and achieves regret $\delta > \sqrt{\frac{16\log^2 n}{T}}$ in the random-order model (assuming the number of mistakes M made by the best expert is known)

- Remove the assumption that M is known by using random-order property plus prefix of the stream to estimate M

Removing the Assumption on M

- Do a binary search for $\frac{M}{T}$ with γ as the running estimate
- Proceed through $\ell = 2\log\frac{1}{\delta}$ epochs, each of length $\frac{\delta T}{\ell}$
- Run previous algorithm on with estimated cost $\gamma \cdot \frac{\delta T}{\ell}$ and target regret $O(1)$ until we have a $(1 + O(\delta))$-approximation of $\frac{M}{T}$ by γ

- Since regret is lower, space usage increases by a factor of $O(\ell)$ for $\delta \leq \frac{M}{T}$

Summary of Random-Order Model

- Given $\delta > \sqrt{\frac{16 \log^2 n}{T}}$, there exists an algorithm in the random-order model that uses achieves expected regret δ and uses $O\left(\frac{n}{\delta^2 T} \log^2 n\right)$ space

- Generalizes to other sequential prediction algorithms!

Summary of Results

- Any algorithm that achieves $\delta < \frac{1}{2}$ regret with probability at least $\frac{3}{4}$ must use $\Omega\left(\frac{n}{\delta^2 T}\right)$ space
- There exists an algorithm that uses $O\left(\frac{n}{\delta^2 T}\log^2 n\right)$ space in the random-order model
- For $M < \frac{\delta^2 T}{1280 \log^2 n}$, there exists an algorithm that uses $\tilde{O}\left(\frac{n}{\delta T}\right)$ space and achieves regret δ
- If the cost is between $[0, \rho]$, the regret is $\rho\delta$ for both models

- Questions: tight bounds for arbitrary order streams?

  how general is this framework beyond the experts problem?