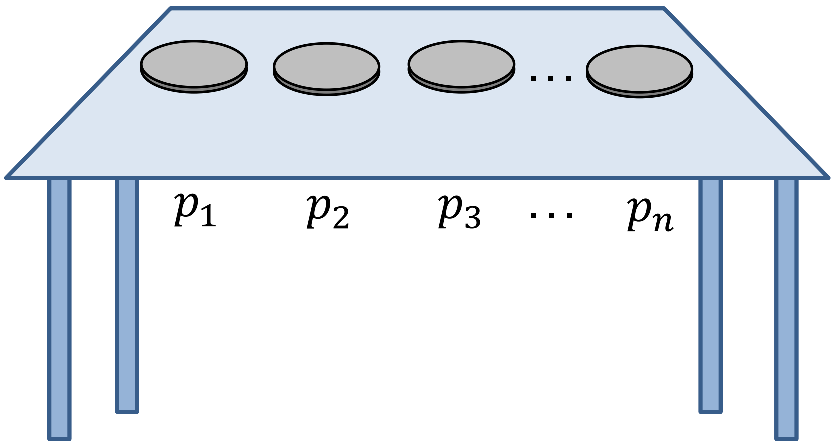
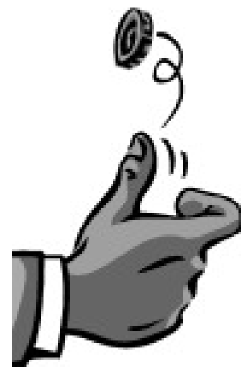
Finding a most biased coin with fewest flips

Karthik Chandrasekaran (Simons Postdoc, Harvard)

Richard Karp (University of California, Berkeley)

The Coin-Toss Problem





- Given: n coins with bias probability p_i for coin i and $\delta, \epsilon > 0$
- Allowed operation: Toss and note down history
- Goal: Find a coin i with large bias $\Pr(p_i \geq p^* - \epsilon |\ i \text{ is output}) \geq 1 - \delta$
- Qn: Is there a strategy that minimizes the expected number of tosses needed?
- Indifference zone assumption: top two bias probabilities differ by at least ϵ

Related work

- Number of tosses for non-adaptive $\leq \left(\frac{4n}{\epsilon^2}\right) \log\left(\frac{n}{\delta}\right)$
- [EMM '02] Upper bound: Adaptive Algorithm
  - Number of tosses $= O\left(\frac{n}{\epsilon^2} \log\left(\frac{1}{\delta}\right)\right)$
- [MT '04] Lower bound:
  - There exist probabilities $p_1, \ldots, p_n$ such that the number of tosses for any adaptive strategy $=$ $\Omega\left(\frac{n}{\epsilon^2} \log\left(\frac{1}{\delta}\right)\right)$

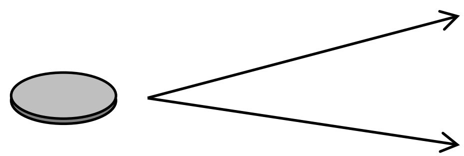
Question: Better algorithm to address the constant factor gap?

A Decision-Theoretic Perspective

- Given: a history of toss outcomes
- Determine the coin to toss in the next step so that the expected future number of tosses to find a large bias coin is minimized
- Seeking optimum decision in each step
  - Does such a strategy even exist?
  - If it does, can we implement it efficiently?

Problem Setting

- Coins are of two types
  - Most biased: $p_i = p + \epsilon$ ⟵ Heavy coin
  - Second-most biased: $p_i = p - \epsilon$ ⟵ Non-heavy coin
- Infinite supply of coins with probabilistic prior



Heavy with probability α

Non-heavy with probability $1 - \alpha$

- Algorithm is allowed to toss coins adaptively
- Output a coin i s.t.

$$\Pr(\text{Coin } i \text{ is heavy} \mid \text{history}(i)\,) \geq 1 - \delta$$

Results

An optimal algorithm: minimizes the expected number of tosses

- ➢ Decision Theoretic Optimal Strategy: In each step, the strategy picks a coin so that the expected future number of tosses is minimized
  - Can start with an arbitrary history for a finite collection of the coins
- ➢ Expected number of coin tosses $\leq \left(\frac{32}{\epsilon^2}\right)\left(\frac{1-\alpha}{\alpha} + \log\left(\left(\frac{1-\delta}{\delta}\right)\left(\frac{1-\alpha}{\alpha}\right)\right)\right)$

The Strategy

Likelihood Ratio

- For a coin i with $\text{history}(i) = \big(\#\text{heads}(i), \#\text{tails}(i)\big)$, define

$$L_i := \frac{\Pr(\text{history}(i) | \text{Coin } i \text{ is heavy})}{\Pr(\text{history}(i) | \text{Coin } i \text{ is non−heavy})}$$

$$= \left(\frac{p+\epsilon}{p-\epsilon}\right)^{\#\text{heads}(i)} \left(\frac{1-p-\epsilon}{1-p+\epsilon}\right)^{\#\text{tails}(i)}$$

Observation: Given the $\text{history}(i) = \big(\#\text{heads}(i), \#\text{tails}(i)\big)$ for coin i

$\Pr\big(\text{Coin } i \text{ is heavy} | \text{history}(i)\big) \geq 1-\delta$ if and only if $L_i \geq \left(\frac{1-\delta}{\delta}\right)\left(\frac{1-\alpha}{\alpha}\right)$

Algorithm

- Initialize $L_i = 1$ for every coin i
- While $L_i < \left(\frac{1-\delta}{\delta}\right)\left(\frac{1-\alpha}{\alpha}\right)$ for every coin i:
  - Toss coin i for which L_i is largest
  - Update L_i based on toss outcome:

$$L_i \leftarrow \begin{cases} L_i\left(\dfrac{p+\epsilon}{p-\epsilon}\right) & \text{if outcome is head} \\ L_i\left(\dfrac{1-p-\epsilon}{1-p+\epsilon}\right) & \text{if outcome is tail} \end{cases}$$

- Output coin i with largest L_i

Correctness Probability

Observation: Given the $\text{history}(i) = \left(\#\text{heads}(i), \#\text{tails}(i)\right)$ for coin i

$\Pr\left(\text{Coin } i \text{ is heavy} | \text{history}(i)\right) \geq 1 - \delta$ if and only if $L_i \geq \left(\frac{1-\delta}{\delta}\right)\left(\frac{1-\alpha}{\alpha}\right)$

If coin i is output by the algorithm, then $L_i \geq \left(\frac{1-\delta}{\delta}\right)\left(\frac{1-\alpha}{\alpha}\right)$

$$\Rightarrow \Pr\left(\text{Coin } i \text{ is heavy} | \text{history } (i)\right) \geq 1 - \delta$$

Optimality

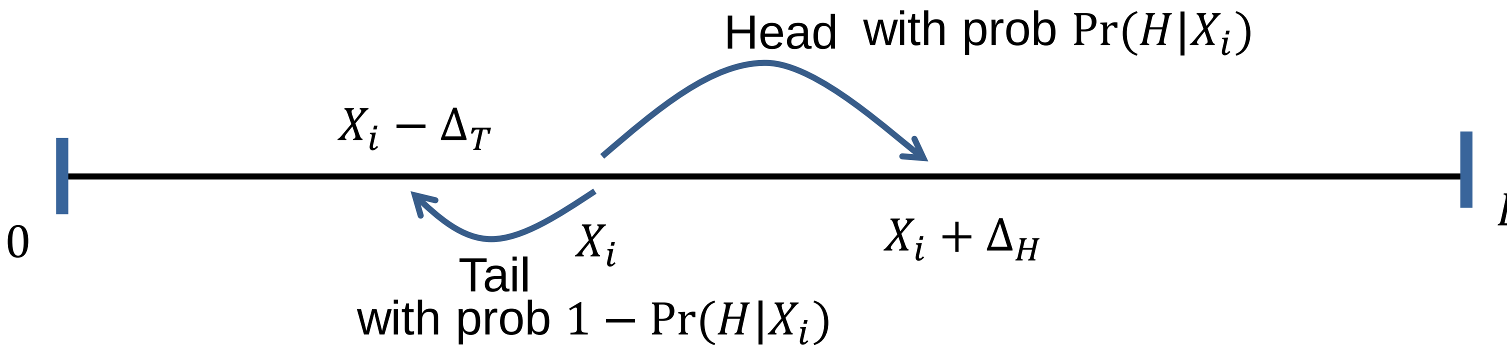
An alternate view of the algorithm

$$X_i := \log L_i$$

$$\Delta_H := \log\left(\frac{p+\epsilon}{p-\epsilon}\right)$$

$$B := \log\left(\left(\frac{1-\delta}{\delta}\right)\left(\frac{1-\alpha}{\alpha}\right)\right)$$

$$\Delta_T := \log\left(\frac{1-p+\epsilon}{1-p-\epsilon}\right)$$

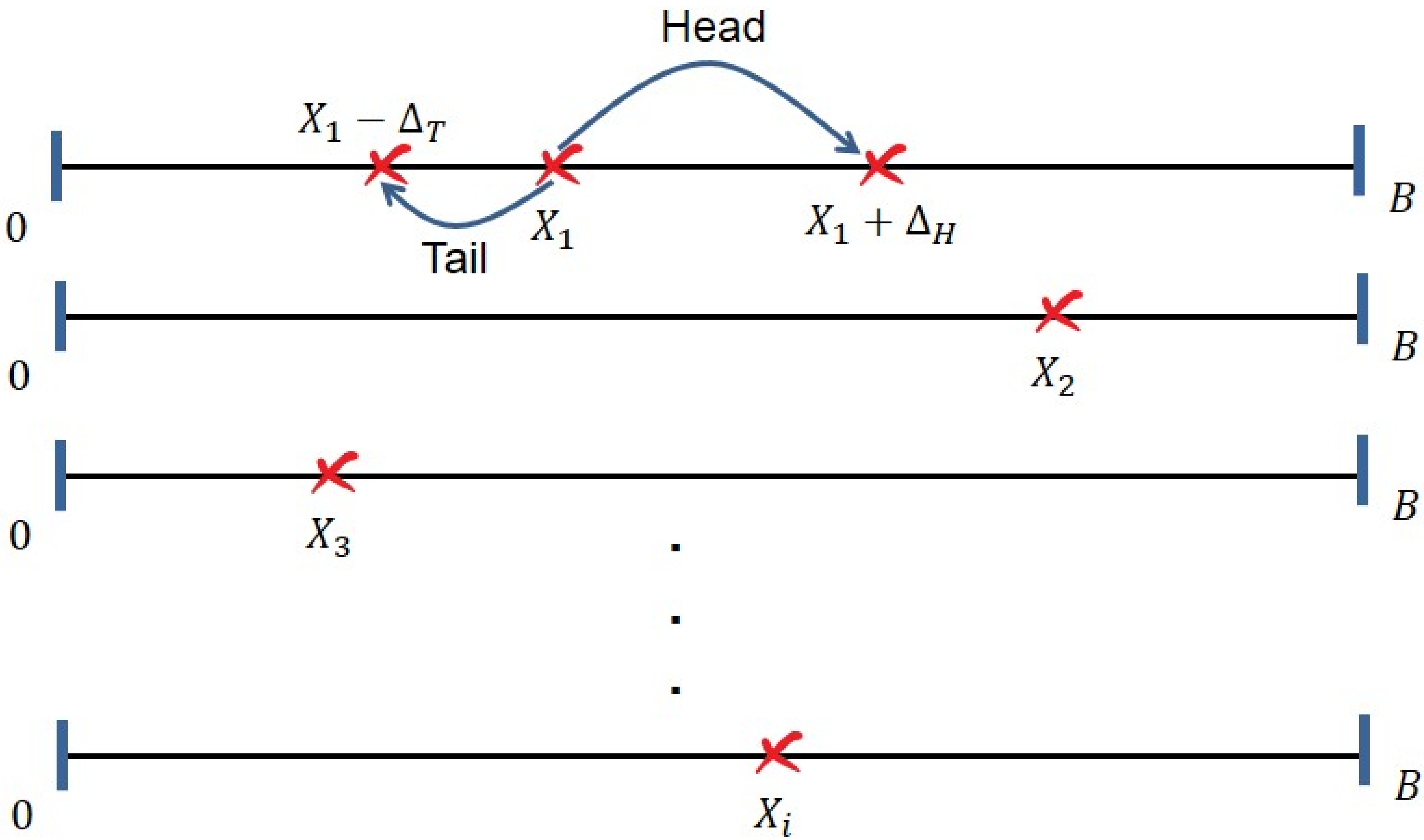


$$\Pr(H|X_i) = \Pr(i \text{ is heavy}|X_i)\Pr(H|i \text{ is heavy})$$

$$+\Pr(i \text{ is non-heavy}|X_i)\Pr(H|i \text{ is non-heavy})$$

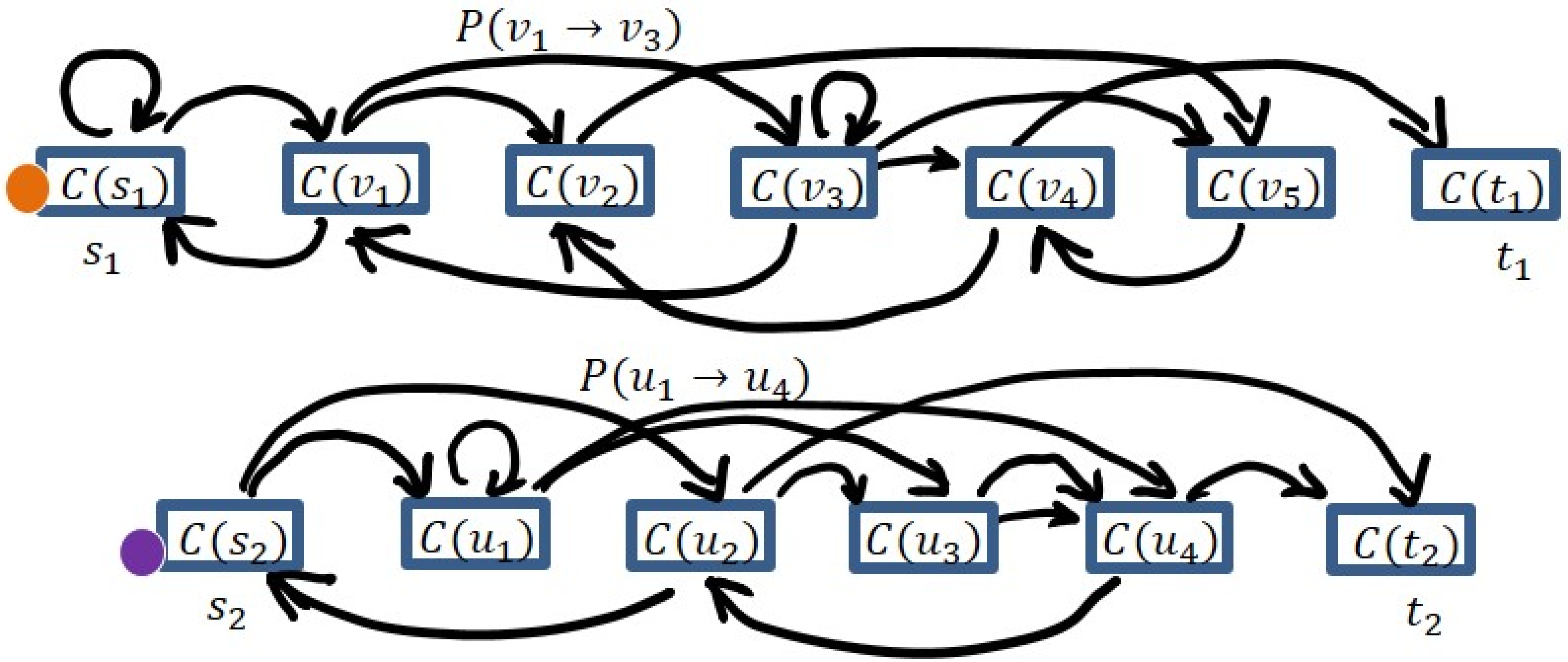
$$= \frac{\alpha e^{X_i}}{\alpha e^{X_i} + (1-\alpha)}(p+\epsilon) + \frac{(1-\alpha)}{\alpha e^{X_i} + (1-\alpha)}(p-\epsilon)$$

An alternate view of the algorithm



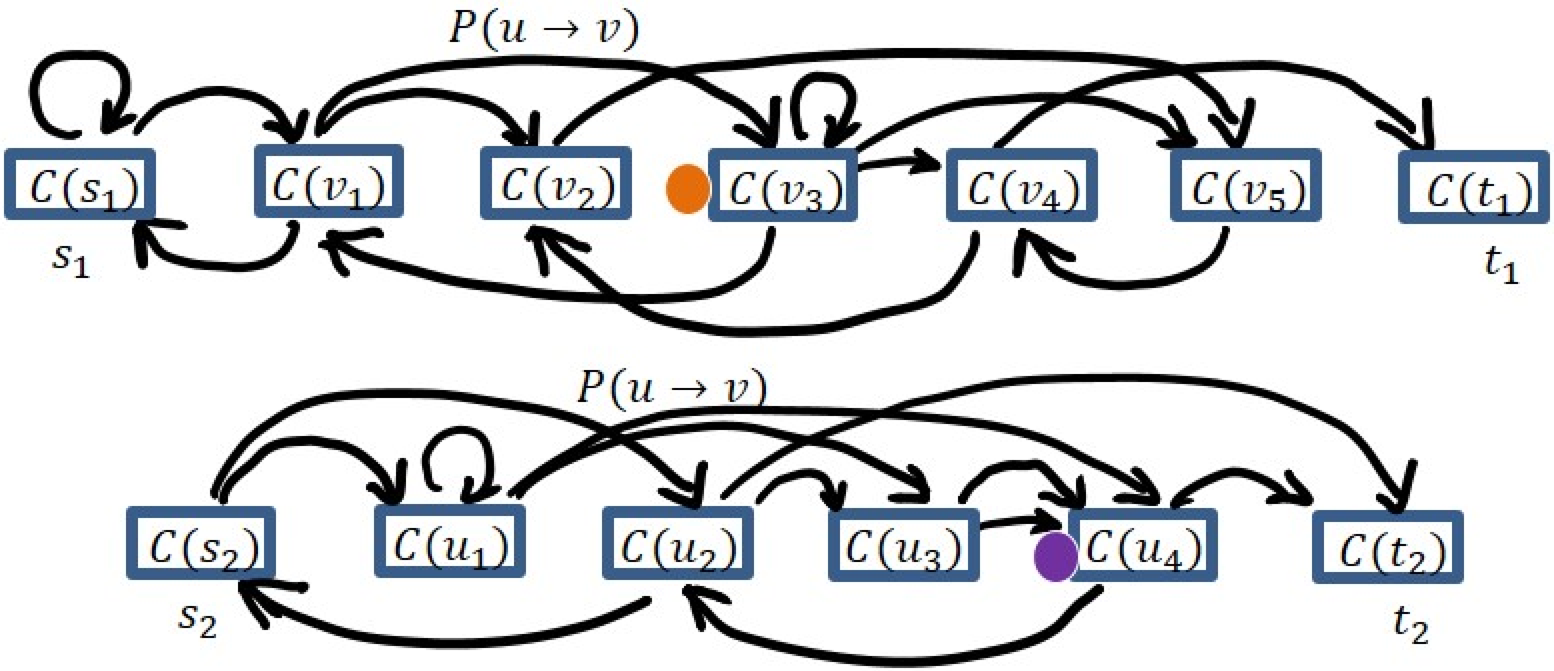
Multi-token Markov Game

Multi-token Markov Game



- Cost of the game $:= \min_{strategy\ \Pi} \mathbb{E}\big(cost(\Pi)\big)$
- Strategy can be randomized

Optimal Strategy [DTW '03]



- Optimal strategy: pick the token in a state with the least *grade*

$$\text{grade}: States \to \mathbb{R}$$

Optimality for our Markov Game

- Greedy strategy: Toss the coin with max likelihood
- Goal: Show that the greedy strategy is optimal
- State space: (all real values $\leq B$) corresponds to log-likelihoods
- Lemma: grade is non-increasing as a function of log-likelihood
- [DTW '03]: Picking the token with the least grade is optimal

$\Rightarrow$ Tossing the coin with maximum log-likelihood is optimal

Open Questions

- Three types of coins
  - Bias probabilities: $(p + \epsilon, p, p - \epsilon)$
  - Infinite supply of coins with probabilistic prior:

$$(\alpha_1, \alpha_2, 1 - \alpha_1 - \alpha_2)$$

- Two types of coins, but dependent prior
  - n coins containing exactly one heavy coin
  - Other prior models?

Questions?

Thank you