Mechanism Design in Large Games: Incentives and Privacy.

Aaron Roth

Joint work with

Michael Kearns, Mallesh Pai, Jon Ullman

Consider the following scenario.

GPS assisted navigation.

- You type in your destination, Google tells you a strategy for getting there.
- What strategy should Google compute?
- Right now, a best response.

Consider the following scenario.

GPS assisted navigation.

But what if everyone uses Google Navigation?

Now Google *creates* traffic.

Consider the following scenario.

GPS assisted navigation.

But what if everyone uses Google Navigation?

Could compute a solution to minimize average congestion…

Consider the following scenario.

GPS assisted navigation.

But what if everyone uses Google Navigation?

But this leaves the door open to a competing GPS service.

Consider the following scenario.

GPS assisted navigation.

But what if everyone uses Google Navigation?

Instead, Google should compute an *equilibrium*.

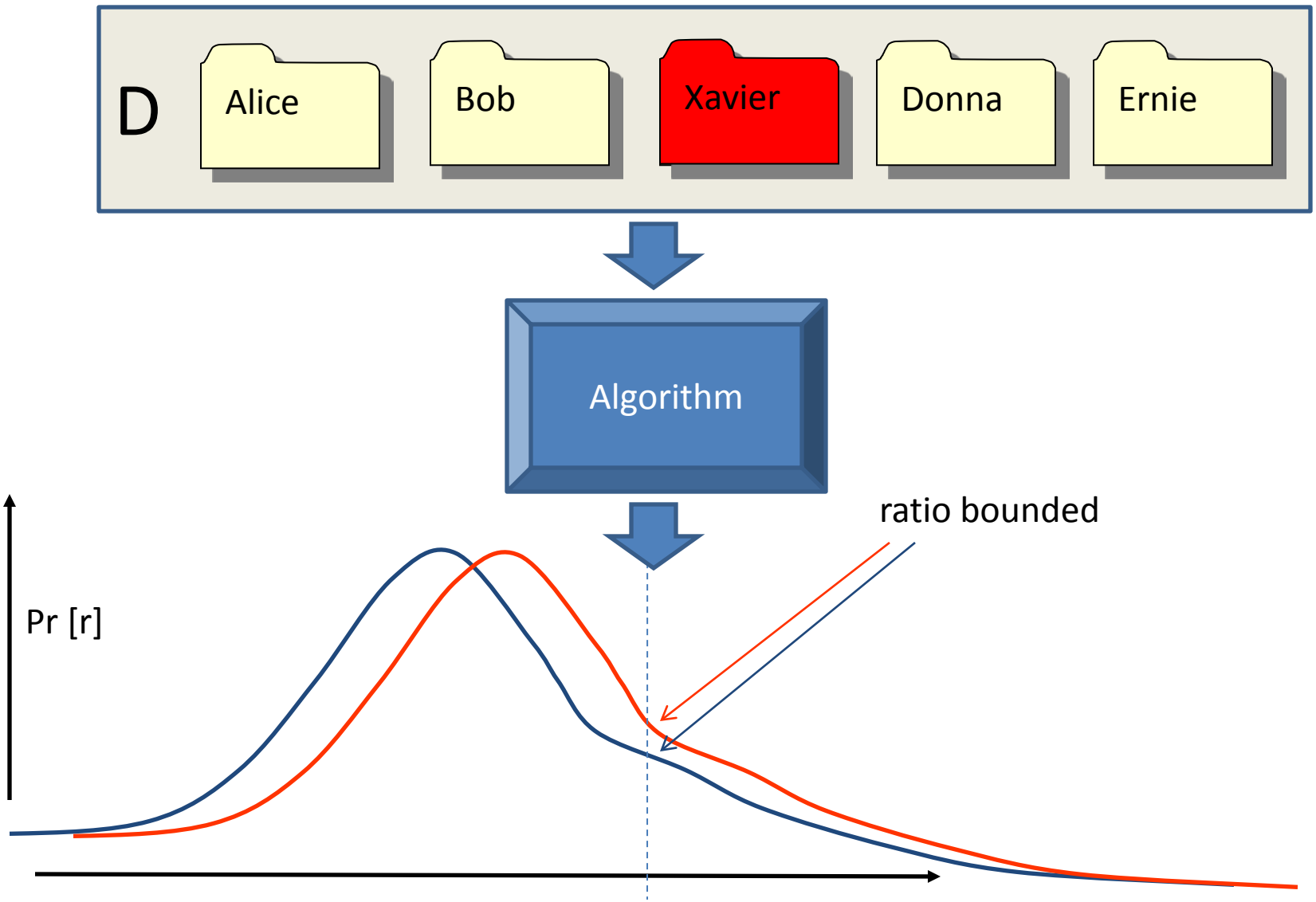
Two Concerns

1. Privacy!
   - Alice's directions depend on my input!
   - Can she learn about where I am going?

Two Concerns

2. Incentives!
   - Alice's directions depend on my input!
   - Can I benefit by misreporting my destination?
     - Causes Google to compute an equilibrium to the wrong game.
     - Might reduce traffic along the route I really want.

Both Addressed by (Differential) Privacy



Both Addressed by (Differential) Privacy

An algorithm A with domain X and range R is ϵ-private if for every utility function $u: R \to \mathbb{R}$ and for every pair of databases $D, D' \subset X$ differing in a single record:

$$\mathbb{E}_{x \sim A(D)}[u(x)] \leq (1 + \epsilon) \cdot \mathbb{E}_{x \sim A(D')}[u(x)]$$

Game Theoretic Implications

- [MT07] A mechanism that is ϵ-private is also ϵ-approximately truthful.
- Simple Corollary:

A mechanism which computes an α-approximate equilibrium while preserving ϵ-privacy makes truthful reporting, followed by suggested play, an $(\epsilon + \alpha)$-Nash equilibrium.

Most interesting when $(\epsilon + \alpha) \to 0$

What can we hope for?

We shouldn't expect to be able to privately solve "small" games.

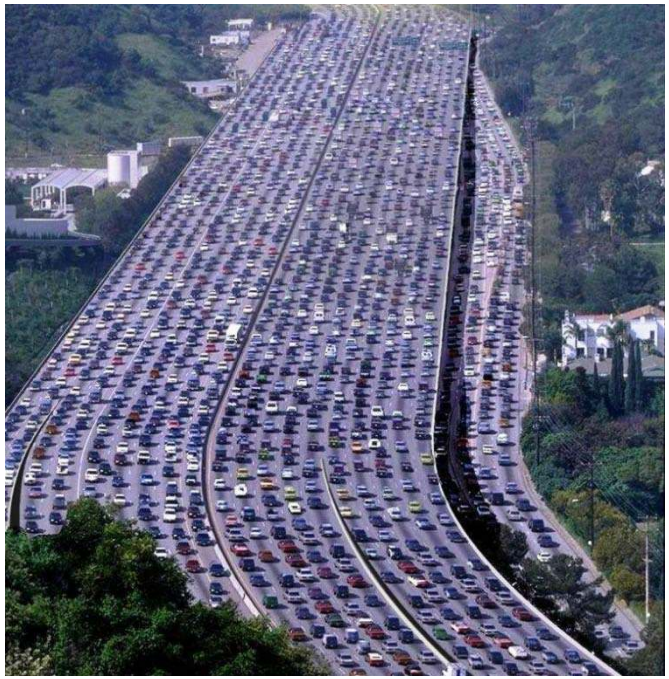
(Alice's best response reveals Bob's action, and therefore potentially his utility function)

		Woman: Baseball	Woman: Ballet
Man	Baseball	(3, 2)	(1, 1)
	Ballet	(0, 0)	(2, 3)

What can we hope for?

Instead, focus on *large* games.

(In which no player has a substantial impact on the utility of others…)



Large Games

A game is Δ-large if for all players $i \neq j \in [n]$, for all action profiles $s \in [k]^n$ and for all pairs of actions $s_j, s_j' \in [k]$:

$$\left|u_i(s_j, s_{-j}) - u_i(s_j', s_{-j})\right| \leq \Delta$$

- Think of $\Delta = o(1)$. In this talk, $\Delta = O\left(\frac{1}{n}\right)$.
- Your action can have a large effect on your own payoff, but not on that of others.

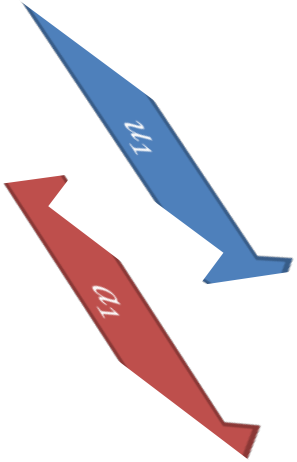
What are our inputs and outputs?

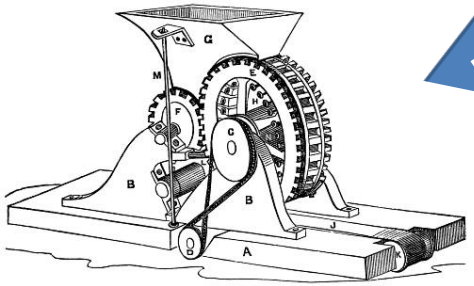
- Input: n utility functions $u_i : [k]^n \to [0,1]$
- Output: n actions $a_i \in [k]$ which are draws from an approximate correlated equilibrium.
  - In fact, we prove that privacy would still be preserved even if we output the full marginal distribution.

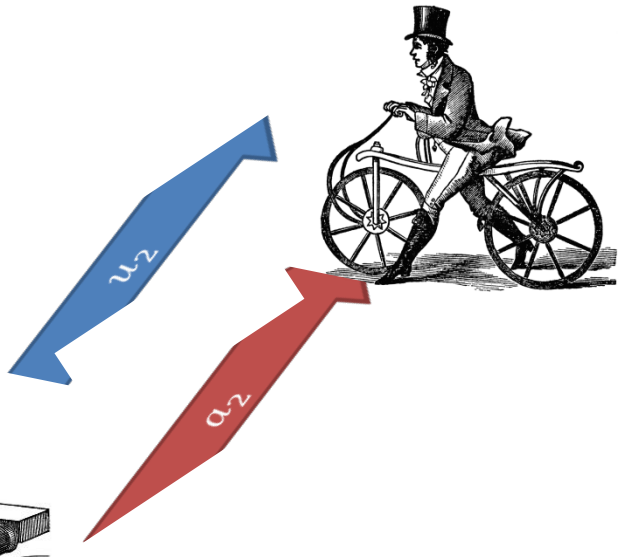
What are our inputs and outputs?

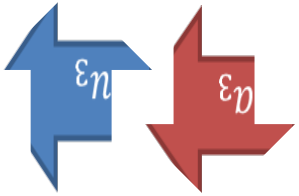
- Input: n utility functions $u_i : [k]^n \to [0,1]$
- Output: n actions $a_i \in [k]$ which are draws from an approximate correlated equilibrium.
  - Since a_i can be highly sensitive to u_i, can't just publish the whole output…

What are our inputs and outputs?









What are our inputs and outputs?

We require that for all players i, the joint distribution over the actions a_j *for all* $j \neq i$ is differentially private in u_i.

- i.e. privacy is preserved even if all *other* players collude and share their outputs, so long as you don't share yours.

So what can we do?

First, what we can't do:

Theorem: No differentially private mechanism can compute an α-approximate coarse correlated equilibrium for $\alpha = \Omega\left(\frac{1}{\sqrt{n}}\right)$, even for games with only $k = 2$ actions.

Proof Idea

- Reduce to reconstruction lower bounds for answering subset-sum queries on boolean valued databases, due to [DinurNissim03], [DworkMcsherryTalwar07] [DworkYekhanin08].
- "Any private mechanism which answers $O(n)$ 'subset sum' queries over n bits must have error $\Omega(\sqrt{n})$"

Proof Idea

- Answer the queries with a game.

Data Players

Query Players

Proof Idea

- Answer the queries with a game.

Data Players

Two actions: $\{0, 1\}$

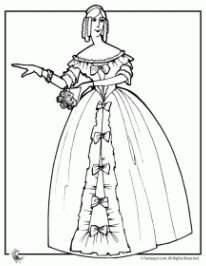
Parameterized by a bit b_i.

$$u_i(s) = \begin{cases} 1 & if\ s_i = b_i \\ 0 & otherwise \end{cases}$$

Proof Idea

- Answer the queries with a game.

Query Players



$\frac{1}{\alpha}$ actions: $\{0, \alpha, 2\alpha, \ldots, 1\}$

Parameterized by a subset of data players T.

$$u_i(s) = 1 - \left| s_i - \frac{1}{T} \sum_{j \in T} s_j \right|$$

Proof Idea

- From every α-approximate CCE, can recover α-approximate answers to all of the query players' subset sum queries.
  - Recall that even if the query players share their equilibrium strategies, the privacy of all the data players is still preserved.
  - Lower bound now follows from [DN03,DMT07,DY08]
  - A little more work can reduce query players action set to 2.

So what can we do?

Theorem: There exists a computationally efficient algorithm which computes an α-approximate CE of a large game with n players and k actions, while preserving ϵ-privacy for:

$$\alpha = O\left(\frac{k^{\frac{3}{2}}}{\epsilon \cdot \sqrt{n}}\right)$$

- Tight for games with $k = O(1)$ actions.

Proof Idea

- Computing a correlated equilibrium can be reduced to approximately answering a small number of numeric valued queries (We'll see this)
- Can use tools from the privacy literature to do this privately.

So what can we do?

Theorem: There exists a computationally *inefficient* algorithm which computes an α-approximate CE of a large game with n players and k actions and T types, while preserving ϵ-privacy for:

$$\alpha = \mathrm{O}\left(\frac{\log(\mathrm{k}) \cdot \log(T)^{3/2}}{\epsilon \cdot \sqrt{\mathrm{n}}}\right)$$

- Nontrivial even for exponential k.

Proof Idea

- Same as before, but use more sophisticated methods [RR10,HR10] to estimate utilities privately. With less noise.
  - Less computationally efficient.

Approximately Truthful Equilibrium Selection

- Recall that everyone truthfully reporting their utility function, and then taking the suggested equilibrium action from an $(\epsilon + \alpha)$-Nash equilibrium
- Choosing ϵ optimally, we get…

Approximately Truthful Equilibrium Selection

Theorem: In any large game, there is a computationally efficient, η-approximately truthful equilibrium selection mechanism for:

$$\eta = O\left(\frac{k^{3/4}}{n^{1/4}}\right)$$

Approximately Truthful Equilibrium Selection

Theorem: In any large game, there is a computationally inefficient, η-approximately truthful equilibrium selection mechanism for:

$$\eta = O\left(\frac{\sqrt{\log(k) \cdot \log(T)^{3/2}}}{n^{1/4}}\right)$$

- Approaches exact truthfulness as the population grows.
- "Equilibrium selection is a problem of small games"

Reducing Equilibrium Computation to Estimating A Small Number of Numeric Queries.

Using “expert” advice

Say we want to predict the stock market.

- We solicit N “experts” for their advice. (Will the market go up or down?)
- We then want to use their advice somehow to make our prediction. E.g.,

Expt 1	Expt 2	Expt 3	neighbor’s dog	truth
down	up	up	up	up
down	up	up	down	down
…	…	…	…	…

Can we do nearly as well as best in hindsight?

[“expert” ´ someone with an opinion. Not necessarily someone who knows anything.]

Simpler question

- We have N "experts".
- One of these is perfect (never makes a mistake). We just don't know which one.
- Can we find a strategy that makes no more than lg(N) mistakes?

Answer: sure. Just take majority vote over all experts that have been correct so far.

➢Each mistake cuts # available by factor of 2.

➢Note: this means ok for N to be very large.

"halving algorithm"

Using “expert” advice

But what if none is perfect? Can we do nearly as well as the best one in hindsight?

Strategy #1:

- Iterated halving algorithm. Same as before, but once we've crossed off all the experts, restart from the beginning.
- Makes at most $\lg(N)[OPT+1]$ mistakes, where OPT is #mistakes of the best expert in hindsight.

Seems wasteful. Constantly forgetting what we've “learned”. Can we do better?

Weighted Majority Algorithm

Intuition: Making a mistake doesn't completely disqualify an expert. So, instead of crossing off, just lower its weight.

Weighted Majority Alg:

- Start with all experts having weight 1.
- Predict based on weighted majority vote.
- Penalize mistakes by cutting weight in half.

					prediction	correct
weights	1	1	1	1		
predictions	Y	Y	Y	N	Y	Y
weights	1	1	1	.5		
predictions	Y	N	N	Y	N	Y
weights	1	.5	.5	.5		

Analysis: do nearly as well as best expert in hindsight

- M = # mistakes we've made so far.
- m = # mistakes best expert has made so far.
- W = total weight (starts at N).

- After each mistake, W drops by at least 25%.

So, after M mistakes, W is at most $N(3/4)^M$.

- Weight of best expert is $(1/2)^m$. So,

$$\begin{aligned} (1/2)^m &\leq N(3/4)^M \\ (4/3)^M &\leq N2^m \\ M &\leq 2.4(m + \lg N) \end{aligned}$$

constant ratio

Randomized Weighted Majority

2.4(m + lg N) not so good if the best expert makes a mistake 20% of the time. Can we do better? Yes.

- Instead of taking majority vote, use weights as probabilities. (e.g., if 70% on up, 30% on down, then pick 70:30) Idea: smooth out the worst case.
- Also, generalize ½ to 1- ε.

Solves to: $M \leq \frac{-m \ln(1-\varepsilon) + \ln(N)}{\varepsilon} \approx (1+\varepsilon/2)m + \frac{1}{\varepsilon}\ln(N)$

M = expected #mistakes

$M \leq 1.39m + 2 \ln N \quad \leftarrow \varepsilon = 1/2$

$M \leq 1.15m + 4 \ln N \quad \leftarrow \varepsilon = 1/4$

$M \leq 1.07m + 8 \ln N \quad \leftarrow \varepsilon = 1/8$

Analysis

- Say at time t we have fraction F_t of weight on experts that made mistake.
- So, we have probability F_t of making a mistake, and we remove an εF_t fraction of the total weight.
  - $W_{final} = N(1-\varepsilon F_1)(1 - \varepsilon F_2)...$
  - $\ln(W_{final}) = \ln(N) + \sum_t [\ln(1 - \varepsilon F_t)] < \ln(N) - \varepsilon \sum_t F_t$

    (using $\ln(1-x) < -x$)

    $= \ln(N) - \varepsilon M.$ ($\sum F_t$ = E[# mistakes])
- If best expert makes m mistakes, then $\ln(W_{final}) > \ln((1-\varepsilon)^m)$.
- Now solve: $\ln(N) - \varepsilon M > m \ln(1-\varepsilon)$.

$$M \leq \frac{-m \ln(1-\varepsilon) + \ln(N)}{\varepsilon} \approx (1+\varepsilon/2)m + \frac{1}{\varepsilon}\log(N)$$

Summarizing

- E[# mistakes] < $(1+\varepsilon)m + \varepsilon^{-1}\log(N)$.

- If set $\varepsilon=(\log(N)/m)^{1/2}$ to balance the two terms out and get bound of E[mistakes] = $m+2(m\log N)^{1/2}$
- Since m < T, this is at most $m + 2(T\log(N))^{1/2}$.
- $\frac{M}{T} < \frac{m}{T} + \sqrt{\frac{\log(N)}{T}}$

What if we have N options, not N predictors?

- We're not combining N experts, we're choosing one. Can we still do it?
- Nice feature of RWM: can still apply.
  - Choose expert i with probability $p_i = w_i/W$.
  - Still the same algorithm!
  - Can apply to choosing N options, so long as costs are {0,1}.
  - What about costs in [0,1]?

What if we have N options, not N predictors?

What about costs in [0,1]?

- If expert i has cost c_i, do: $w_i = w_i(1-c_i\varepsilon)$.
- Our expected cost = $\sum_i c_i w_i / W$.
- Amount of weight removed = $\varepsilon \sum_i w_i c_i$.
- So, fraction removed = ε * (our cost).
- Rest of proof continues as before…

What does this have to do with computing equilibria?

- It is natural to use the weighted majority algorithm to play a game.
  - Identify experts with actions, payoffs with utilities.
- If all players use WM algorithm to play for T rounds, we end up with profiles: $s^1, \ldots s^T$ such that for each player i and action a_i:

$$E_{t\sim[T]}[u_i(s^t)] \geq E_{t\sim[T]}\left[u_i(a_i, s^t{}_{-i})\right] - \sqrt{\frac{\log(N)}{T}}$$

What does this have to do with computing equilibria?

- Taking $T = \frac{\ln(N)}{\alpha^2}$ and we get:

$$E_{t\sim[T]}[u_i(s^t)] \geq E_{t\sim[T]}[u_i(a_i, s^t{}_{-i})] - \alpha$$

- An α-approximate "Coarse Correlated Equilibrium"
- A little more work gets convergence to correlated equilibrium.

Computing an Equilibrium with Very Little Information

- The game matrix of an n player k action game has size $\approx k^n$.
- Yet we can compute an α-approximate correlated equilibrium by communicating only $\approx k \frac{\log k}{\alpha^2}$ utilities per player.

Computing an Equilibrium with Very Little Information

- These reported utilities need not be exact…
- Recall what we are bounding is:

$$\max_{a_i} \sum_{t=1}^{T} u_i\left(a_i, s^t{}_{-i}\right) - \sum_{t=1}^{T} u_i(s^t)$$

- What if the algorithm instead observes payoff estimates $\hat{u}_i$ such that for all a_i:

$$\frac{1}{T}\left|\sum_{t=1}^{T} \hat{u}_i\left(a_i, s^t{}_{-i}\right) - \sum_{t=1}^{T} u_i\left(a_i, s^t{}_{-i}\right)\right| \leq \beta$$

Computing an Equilibrium with Very Little Information

- Then, we get a sequence of action such that:

$$E_{t\sim[T]}[u_i(s^t)] \geq E_{t\sim[T]}\left[u_i(a_i, s^t{}_{-i})\right] - \alpha - 2\beta$$

- i.e. we get an $(\alpha + 2\beta)$-approximate equilibrium.
- In reality, both α, β will be a function of T.
  - Increasing T decreases α (as we saw) but increases β.
  - Can pick T to optimize the tradeoff…

Briefly…

- We took the perspective of mechanism designers:
  - We simulate play of the game to compute a solution
  - We add noise explicitly.

Briefly…

- Instead, can think of the noise as inherent to the interaction and study the equilibria of the repeated game.
  - Even in the infinitely repeated game, if the noise rate grows…
    - Or if the noise is constant and the *population* grows
  - … the observed payoffs of each player $j \neq i$ will be differentially private in i's actions.

Briefly

- Then, all of the “Folk Theorem” equilibrium of the repeated game are eliminated.
  - Intuition: If play is privacy preserving, this removes the power to punish deviations.
  - Equilibrium of the repeated game collapse to equilibrium of the single shot game.
- A little noise can improve the “price of anarchy” of the repeated game by arbitrarily large factors.

Open Questions

- Can we get sub-polynomial dependence on k in polynomial time?
- Can we get sub-polynomial dependence on k without dependence on the size of the type space?
- Better equilibrium selection mechanisms via other means?
- What else can privacy say about noise in games?

Open Questions

- Can we get sub-polynomial dependence on k in polynomial time?
- Can we get sub-polynomial dependence on k without dependence on the size of the type space?
- Better equilibrium selection mechanisms via other means?
- What else can privacy say about noise in games?

Thank You!