Learning to Play General-Sum Games Against Multiple Boundedly Rational Agents

Eric Zhao[1,2], Alexander R. Trott[1], Caiming Xiong[1], Stephan Zheng[1]

[1]Salesforce Research. Palo Alto, California, USA
[2]University of California, Berkeley. Berkeley, California, USA

Abstract

We study the problem of training a principal in a multi-agent general-sum game using reinforcement learning (RL). Learning a robust principal policy requires anticipating the worst possible strategic responses of other agents, which is generally NP-hard. However, we show that no-regret dynamics can identify these worst-case responses in poly-time in smooth games. We propose a framework that uses this policy evaluation method for efficiently learning a robust principal policy using RL. This framework can be extended to provide robustness to boundedly rational agents too. Our motivating application is automated mechanism design: we empirically demonstrate our framework learns robust mechanisms in both matrix games and complex spatiotemporal games. In particular, we learn a dynamic tax policy that improves the welfare of a simulated trade-and-barter economy by 15%, even when facing previously unseen boundedly rational RL taxpayers.

1 Introduction

We study the problem of learning a principal policy in a general-sum game against boundedly rational agents. Learning a robust principal policy requires us to anticipate how these agents may respond to our policy choices, and entails two important challenges (Figure 1). First, the policies we choose induce a sub-game between the other agents, a sub-game which may have infinite equilibria. The policy we choose should perform well regardless of which equilibria the agents respond with. Second, principal policies that perform well against rational agents may not generalize to boundedly rational agents, even if they are only infinitesimally irrational (Pita et al. 2010). Our policy should perform well even if agents act boundedly rational.

We introduce a framework for the reinforcement learning of robust principal policies that address these two challenges. This framework evaluates a potential policy by identifying the worst-case coarse-correlated equilibrium (CCE) of the sub-game the policy induces. Although identifying *worst-case* CCE is generally computationally intractable (Papadimitriou and Roughgarden 2008; Barman and Ligett 2015), we prove that worst-case CCE can efficiently learned in smooth games. Our framework easily extends to identify worst-case approximate CCE. This allows us to learn principal policies that are
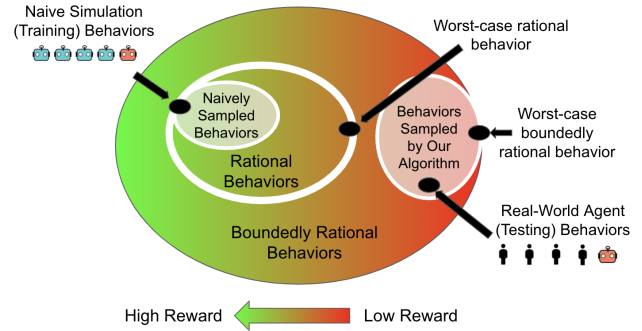
Figure 1: The orange robot denotes our principal, blue robots the agents we train against, and human icons the agents we encounter during test time. Evaluating a policy in a multi-agent game by naively sampling rational responses from other agents, e.g. via multi-agent RL, may lead to overly optimistic reward estimates. We introduce efficient algorithms for adversarially sampling rational responses in smooth games. These algorithms can be extended to sample worst-case boundedly rational responses (bottom-right).

robust to boundedly rational agents, such as agents whose incentives differ from the agents we train against.

Our motivating application is *mechanism design* ("reverse game theory"), where a principal implements the rewards and dynamics (the "mechanism") that other agents optimize for (Myerson 2016). Traditional mechanism design has been limited to problems with a convenient mathematical structure, e.g., simple auctions, where the equilibria behavior of agents can be solved in closed-form. Recent research have pursued computational approaches to mechanism design that evaluate potential mechanisms using agent-based modeling (Holland and Miller 1991; Bonabeau 2002; Duetting et al. 2019) and multi-agent reinforcement learning (MARL) (Zheng et al. 2020). This application of multi-agent learning remains an exciting but understudied topic. We will outline a modern perspective that formalizes automated mechanism design and its robustness concerns as an equilibrium selection problem.

Summary of results.

1. *We motivate a robust learning objective* for finding a principal policy that is robust to agents of differing incentives,

bounded rationality, and reputation. This objective is a multi-follower extension of robust Stackelberg games.

2. *We show the existence of poly-time algorithms* for adversarially sampling the coarse-correlated-equilibria (CCE) of smooth games, proving that multi-follower Stackelberg games can be tractable. This weakens prior findings that learning welfare-maximizing CCE is NP-hard (Papadimitriou and Roughgarden 2008).

3. *We apply our proposed framework to automated mechanism design* problems where multi-agent RL is used to simulate the outcomes of mechanisms. In the spatiotemporal economic simulations used by the AI Economist (Zheng et al. 2020), our framework learns robust tax policies that improve welfare by up to 15% and are robust to previously unseen and boundedly rational agents.

2 Related Work

Finding Equilibria. A goal of multi-agent learning is finding equilibria (or more generally *solution concepts*), i.e., sets of agent policies that are game-theoretically optimal (according to the definition of equilibrium). Prior work has used gradient-based methods, e.g., deep reinforcement learning, to find (approximate) equilibria with great success in multi-player games such as Diplomacy (Gray et al. 2021) and in training Generative Adversarial Networks (Schäfer, Zheng, and Anandkumar 2020). However, learning *robust* principals (or mechanisms) in multi-agent general-sum games is an open problem: it requires evaluating strategies against worst-case sub-game equilibria, which is computationally hard.

Stackelberg Games. Our principal can be seen as a Stackelberg *leader*; the other agents as Stackelberg *followers*. Stackelberg games have had real-world success in security games used in airports (Pita et al. 2009) and anti-poaching defense (Nguyen et al. 2016), for example. However, Stackelberg analysis is typically limited to a single rational follower or assuming that the followers do not strategically interact, e.g., as in multi-follower security games (Korzhyk, Conitzer, and Parr 2011). In contrast, we consider more general settings with multiple followers who may strategically interact with one another. In these settings, finding multi-follower "best-responses" is a computationally hard equilibria search and an open problem (Barman and Ligett 2015; Basilico et al. 2020) for which our work provides a tractable perspective.

Our approach to modeling uncertainty about followers is inspired by *Strict Uncertainty Stackelberg Games* that assume a worst-case choice of a follower's utility function (Pita et al. 2010, 2012; Kiekintveld, Islam, and Kreinovich 2013). Similarly, *Bayesian Stackelberg Games* assume a prior over a space of possible follower utility functions (Conitzer and Sandholm 2006). We extend this by considering a general setting with multiple, possibly interacting, followers.

Robustness. Our work is also related in spirit to prior work on robust reinforcement learning, which we discuss further in Appendix C. We will refer to our notion of robustness as *strategic robustness* to contrast it from non-game-theoretic notions of robustness to, for example, noisy observations (Morimoto and Doya 2000). Strategic robustness also differs

from the topic of "robust game theory", which studies the equilibria that arise when all players act robustly to some uncertainty about game parameters (Aghassi and Bertsimas 2006). Hereafter, we will simply refer to our notion of strategic robustness as "robustness" for brevity.

3 Problem Formulation

Notation. Bold variables are vectors of size n, with each component corresponding to an agent $i = 1, \ldots, N$. For example, $\boldsymbol{a} \in A_{1:n}$ denotes an action vector over all agents except our principal, agent 0. $\boldsymbol{a}_{-i}$ denotes the profile of actions chosen by all agents except agents 0 and i.

Setup. We consider a general-sum game G with $N + 1$ agents. Our principal, or "ego agent", is index $i = 0$ and the other agents are $i = 1, \ldots, N$. A_i denotes the set of m_i actions available to agent i, and $m = \sum_{i=1}^{N} m_i$. $P(A_i)$ is the set of probability distributions over action set A_i. The *joint action set* is $A_{1:n} := \prod_{i \in [1, \ldots, N]} A_i$. $P(A_{1:n})$ denotes the set of joint distributions over strategy profiles $A_{1:n}$. $P^{\mathrm{prod}}(A_{1:n})$ denotes the set of product distributions over strategy profiles $A_{1:n}$. Every agent $i = 0, \ldots, N$ has a utility function $u_i : A_0 \times A_{1:n} \to \mathbb{R}$ with bounded payoffs. For example, $u_i(a_0, \boldsymbol{a})$ denotes the utility of agent i under action $a_0 \in A_0$ by our principal and actions $\boldsymbol{a} \in A$ by agents $1, \ldots, N$. When a_0 is clear from context, we'll write $u_i(\boldsymbol{a})$, suppressing a_0. We denote expected utility as $\overline{u}_i(x_0, \boldsymbol{x}) := \mathbb{E}_{a_0 \sim x_0, \boldsymbol{a} \sim \boldsymbol{x}} u_i(a_0, \boldsymbol{a})$; again we write $\overline{u}_i(\boldsymbol{x})$ when x_0 is clear from context.

Succinct Games. To derive our complexity results, we will use standard assumptions on our game G so that working with equilibria is not trivially hard (Papadimitriou and Roughgarden 2008). We assume $G := (I, T, U)$ is a succinct game, i.e., has a polynomial-size string representation. Here, I are efficiently recognizable inputs, and T and U are polynomial algorithms. T returns $n, m_0, \ldots, m_N$ given inputs $z \in I$, while U specifies the utility function of each agent. We assume G is of *polynomial type*, i.e., $n, m_0, \ldots, m_N$ are polynomial bounded in $|I|$. We assume that G satisfies the *polynomial expectation property*, i.e., utilities $\overline{u}(x_0, \boldsymbol{x})$ can be computed in polynomial time for product distributions $\boldsymbol{x}$. The latter assumption is known to hold for virtually all succinct games of polynomial type (Papadimitriou and Roughgarden 2008). Without these assumptions, simply evaluating the payoff of a coarse-correlated equilibria can require exponential time. All complexity results in our work, including Theorem 1 and cited results from prior works use these assumptions. Later, we will use additional "smoothness" conditions on G to overcome prior hardness results about succinct games.

Problem. Given a game G, we aim to learn a principal policy $x_0 \in P(A_i)$ that maximizes our principal's expected utility $\overline{u}_0(x_0, \boldsymbol{x})$. Here, $\boldsymbol{x}$ is the strategy that agents in a test environment respond to our policy x_0 with. We will assume access to a training environment for G. In this training environment, we assume access to—potentially inaccurate— estimates of the reward functions of the agents; we will write these estimates as $u_1, \ldots, u_N$. For example, the principal $i = 0$ may be a policymaker setting a tax policy x_0 that

maximizes social welfare u_0. In response, the tax-payers play $\boldsymbol{x}$, choosing whether to work and report income.

Objective In order to formalize our robust learning objective, we must define what uncertainty sets $\boldsymbol{X}$ we want learning guarantees for. A behavioral uncertainty set $\boldsymbol{X}(x_0) \subseteq P(A)$ defines the strategies that the test environment agents may respond to a principal policy x_0 with. For simple agent behaviors, one can use imitation learning or domain knowledge to construct these uncertainty sets. In this work, we will study game-theoretic uncertainty sets, for example, where $\boldsymbol{X}(x_0)$ is the set of rational behaviors (see Section 4). Fixing a choice of $\boldsymbol{X}$, we can write our robust learning objective as:

$$\max_{x_0} \min_{\boldsymbol{x} \in \boldsymbol{X}(x_0)} \overline{u}_0(x_0, \boldsymbol{x}). \tag{1}$$

This is a challenging objective: it features nested optimization and requires searching over behavioral uncertainty sets, which is a non-trivial task in complicated games.

4 Finding Uncertainty Sets for Boundedly Rational Agents

A key challenge in our problem formulation is defining our behavioral uncertainty set $\boldsymbol{X}$. In this section, we will first argue for a uncertainty set of *coarse-correlated equilibria (CCE)*. We will then prove that in *smooth games* we can efficiently approximate worst-case CCE. We will finally propose a more relaxed uncertainty set that yields robustness to boundedly rational agents.

Initial Assumptions Our guiding principle for choosing $\boldsymbol{X}$ is that a robust principal can assume that agents are rational, but should still perform if agents act somewhat irrationally or have incentives that slightly differ from anticipated. We will further assume the following and relax them later.

1. The incentives (reward functions) of the test agents exactly agree with our training environment's estimates.
2. All agents are rational expected-utility maximizers. No agent will settle if they want to unilaterally deviate.
3. We, the principal, can commit to a strategy x_0 and will not react to other agents.

Dominant Strategies. A natural choice of $\boldsymbol{X}$ is to extend Stackelberg equilibria to a multi-follower setting and define $\boldsymbol{X}(x_0)$ as the set of best responses to x_0,

$$\boldsymbol{X}(x_0) = \{\boldsymbol{x} \mid \forall i \in [1, \ldots, n], \tilde{\boldsymbol{x}}_{-i} \in P(A)_{-i} : \\ x_i \in \operatorname{argmax}_{x_i} \overline{u}_i(x_0, x_i, \tilde{\boldsymbol{x}}_{-i})\}. \tag{2}$$

However, this set is only non-empty when all agents have dominant strategies: a strong assumption that rarely holds when followers interact with one another.

Stability-Based Equilibria. We can also define an uncertainty set $\boldsymbol{X}(x_0)$ as the stable equilibria that agents may converge to under our policy x_0. This coincides with Equation 2 when it is non-empty. Formally, let

$$\boldsymbol{X}(x_0) = \{\boldsymbol{x} \in P(A) \mid (x_0, \boldsymbol{x}) \in \text{EQ}\},$$

where natural choices for EQ include mixed Nash equilibria (MNE) in which agents do not coordinate:

$$\text{MNE} = \{\boldsymbol{x} \in P^{\text{prod}}(A) \mid \forall i \in [1, \ldots, N], \tilde{a}_i \in A_i : \\ \mathbb{E}_{\boldsymbol{a} \sim \boldsymbol{x}}[u_i(\boldsymbol{a})] \geq \mathbb{E}_{\boldsymbol{a}_{-i} \sim \boldsymbol{x}_{-i}}[u_i(\tilde{a}_i, \boldsymbol{a}_{-i})]\},$$

or more general coarse-correlated equilibria (CCE):

$$\text{CCE} = \{\boldsymbol{x} \in P(A) \mid \forall i \in [1, \ldots, N], \tilde{a}_i \in A_i : \\ \mathbb{E}_{\boldsymbol{a} \sim \boldsymbol{x}}[u_i(\boldsymbol{x})] \geq \mathbb{E}_{\boldsymbol{a}_{-i} \sim \boldsymbol{x}_{-i}}[u_i(\tilde{a}_i, \boldsymbol{x}_{-i})]\}.$$

Here, coarse-correlated equilibria describe more general joint strategies, such as coordination based on shared information.

Computational Hardness. Unfortunately, optimizing the robustness objective in Equation 1 is neither tractable with MNE nor CCE. Finding the MNE/CCE that minimizes a utility function u_0 is equivalent to the NP-hard problem of finding a MNE/CCE that maximizes a linear social welfare objective ν (Daskalakis, Goldberg, and Papadimitriou 2009; Papadimitriou and Roughgarden 2008); we will set $\nu = -u_0$ for convenience. Beyond maximizing ν, simply finding a CCE that does not minimize ν is NP-hard (Barman and Ligett 2015). Formally, consider the decision problem Γ of determining whether a game G (under our assumptions) admits a CCE $\boldsymbol{x}$ such that the expectation of ν, $\overline{\nu}$, satisfies:

$$\overline{\nu}(\boldsymbol{x}) > \min_{\tilde{\boldsymbol{x}} \in \text{CCE}} \overline{\nu}(\tilde{\boldsymbol{x}}).$$

This problem is NP-hard for some choices of ν, including the social welfare function (Barman and Ligett 2015). For our purposes, this implies that even sampling an approximately worst-case equilibria is intractable. This means it could be impossible to efficiently evaluate our principal's policy as it is intractable to guarantee sampling anything other than uninformative equilibria behavior.

Smooth Games and Tractable Uncertainty Sets

Smooth games offer a workaround to this hardness result.

Definition 4.1 (Smooth Games)**.** A *cost-minimization game* with cost functions c_i and objective C is (λ, μ)-smooth if, for all strategies $\boldsymbol{x}, \boldsymbol{x}^* \in P(A)$,

$$\sum_{i=1}^{N} \mathbb{E}[c_i(x_i^*, \boldsymbol{x}_{-i})] \leq \lambda \cdot \mathbb{E}[C(\boldsymbol{x}^*)] + \mu \cdot \mathbb{E}[C(\boldsymbol{x})].$$

The "robust price of anarchy" (RPOA) is defined $\rho := \frac{\lambda}{1-\mu}$.

In fact, we can sample a CCE that approximately maximizes $\nu = -u_0$ with run-time polynomial in the game size, smoothness (λ_G, μ_G) of the original game G and the smoothness $(\lambda_{\tilde{G}}, \mu_{\tilde{G}})$ of a modified game $\tilde{G}$. Here, $\tilde{G}$ is identical to G except each agent's utility is changed from u_i to ν.

Theorem 1. For succinct n-agent m-action games of polynomial type and expectation property, there exists a $\text{Poly}(1/\epsilon, n, m, \rho)$ algorithm that will find an ϵ-CCE $\boldsymbol{x}$ with

$$\overline{\nu}(\boldsymbol{x}) \geq \frac{y}{\rho} - \epsilon.$$

for any $y \leq \max_{\boldsymbol{x}^* \in \text{CCE}} \overline{\nu}(\boldsymbol{x}^*)$, where $\rho = \frac{\lambda_{\tilde{G}}}{1-\max\{\mu_G, \mu_{\tilde{G}}\}}$.

Proof Sketch of Theorem 1. First, we observe that the problem of finding a ν-maximizing CCE reduces to finding a halfspace oracle that optimizes some modified social welfare (Lemma 1). Similar reductions have been described by (Jiang and Leyton-Brown 2011) and (Barman and Ligett 2015).

Lemma 1. Fix a $0 < y \leq \max_{\boldsymbol{x} \in \text{CCE}} \overline{\nu}(\boldsymbol{x})$. Assume there is a Poly$(1/\epsilon, n, m)$-time halfspace oracle that, given a vector $\beta \in \mathbb{R}_+^{1+m}$ with non-negative components, returns an $\boldsymbol{x} \in P(A)$ such that $\beta v(\boldsymbol{x}) \leq 0$, where

$$v(\boldsymbol{x}) = \begin{bmatrix} \overline{u}_1(1, \boldsymbol{x}_{-1}) - \overline{u}_1(\boldsymbol{x}) \\ \vdots \\ \overline{u}_1(m_1, \boldsymbol{x}_{-1}) - \overline{u}_1(\boldsymbol{x}) \\ \vdots \\ \overline{u}_N(1, \boldsymbol{x}_{-N}) - \overline{u}_N(\boldsymbol{x}) \\ \vdots \\ \overline{u}_N(m_N, \boldsymbol{x}_{-N}) - \overline{u}_N(\boldsymbol{x}) \\ y - \overline{\nu}(\boldsymbol{x}) \end{bmatrix} \quad (3)$$

Then, there is a Poly$(1/\epsilon, n, m)$-time algorithm that returns an ϵ-CCE $\boldsymbol{x}$ with $\overline{\nu}(\boldsymbol{x}) \geq y - \epsilon$.

The halfspace oracle's optimization task can be reduced to optimizing social welfare in a game with a RPOA of ρ, which inherits its smoothness from games G and $\tilde{G}$. This upper bounds the ratio between the smallest objective value ν of a CCE and the largest objective value ν of any strategy. Thus, we can use no-regret dynamics (Cesa-Bianchi and Lugosi 2006; Foster and Vohra 1997; Hart and Mas-Colell 2000) in this smooth game to approximate the half-space oracle.

Lemma 2. Let ρ denote an upper bound on the price of anarchy of $G, \tilde{G}$. There exists a Poly$(1/\epsilon, n, m, \rho)$-time halfspace oracle that, given a vector $\beta \in \mathbb{R}_+^{1+m}$ and $y \leq \rho \max_{\boldsymbol{x} \in \text{CCE}} \overline{\nu}(\boldsymbol{x})$, returns an $\boldsymbol{x} \in P(A)$ such that $\beta v(\boldsymbol{x}) \leq \epsilon$ where v is defined as in Equation 3.

Combining Lemmas 1 and 2 constructs our algorithm. $\square$

Informally, this theorem states that it is tractable to find a CCE that maximizes ν up to the price of anarchy. While this relationship is immediate when ν is the social welfare function (Roughgarden 2015), our result shows that we can prove a similar relationship concerning the optimization of CCE against any linear function. This positive result allows for translation between well-known price-of-anarchy bounds and bounds on the tractability of CCE optimization.

Corollary 1. In linear congestion games, for any linear function ν, we can find, in Poly$(1/\epsilon, n, m)$ time, an ϵ-CCE $\boldsymbol{x}$ such that $\overline{\nu}(\boldsymbol{x}) \geq 0.4 \cdot \max_{\boldsymbol{x} \in \text{CCE}} \overline{\nu}(\boldsymbol{x}) - \epsilon$.

While Barman and Ligett (2015) showed the decision problem Γ of finding a non-trivial CCE is NP-hard in general, our Theorem 1 also shows it is tractable in smooth games.

Corollary 2. The decision-problem Γ is in P for games where $\frac{1}{\rho} \max_{\boldsymbol{x} \in \text{CCE}} \overline{\nu}(\boldsymbol{x}) > \min_{\boldsymbol{x} \in \text{CCE}} \overline{\nu}(\boldsymbol{x})$.

Remarks. These theoretical conclusions suggest that although using equilibria-based uncertainty sets may be intractable in some cases, there is a broad class of common problems where CCE uncertainty sets are reasonable and allow for efficient adversarial sampling. The algorithm we construct in Theorem 1 also enjoys two nice properties. First, it only requires oracle access to utility functions (efficient under polynomial expectation property). Second, in the algorithm's self-play subprocedures, each agent can be trained using only their, and their principal's, utility information.

Weakening Assumptions on Agents and Principal Robustness Algorithm

We now further refine our choice of uncertainty set to ensure generalization to agents that violate our behavioral assumptions. We now switch to weaker assumptions:

1. **Subjective rationality:** At test time, an agent's utility $\tilde{u}_i$ may differ from the anticipated utility u_i (Simon 1976). Many models of subjective rationality, such as Subjective Utility Quantal Response (Nguyen et al. 2013), bound the gap between u_i and $\tilde{u}_i$ as $\|u_i - \tilde{u}_i\|_\infty \leq \gamma_s$ with $\gamma_s > 0$.
2. **Procedural rationality:** An agent may not fully succeed in maximizing their utility (Simon 1976), e.g., they could gain up to $\gamma_p > 0$ utility if they unilaterally deviate.
3. **Myopia:** An agent may possess commitment power or otherwise be non-myopic, factoring in long-term incentives with a discount factor $\gamma_m \in (0, 1)$. This relates to notions of exogenous commitment power, e.g., partial reputation, in Stackelberg games (Kreps and Wilson 1982; Fudenberg and Levine 1989).

These variations represent common forms of bounded rationality. We now show that the sampling scheme we devise for Theorem 1 can be extended to maintain robustness despite these weaker assumptions. We aim to learn strategies x_0 that perform well even when presented with agents possessing these variations. Hence, we aim to use uncertainty sets $\boldsymbol{X}$ that encode such behaviors. The next proposition suggests it suffices to simply relax our uncertainty set $\boldsymbol{X}$ to include more approximate equilibria.

Proposition 1. The uncertainty set $\boldsymbol{X}'$ of (any combination of) agents violating assumptions 1-3 with parameters $\gamma_m, \gamma_s, \gamma_p$ is contained in the set of ε-CCE:

$$\text{CCE}_\varepsilon = \{\boldsymbol{x} \in P(A) \mid \forall i \in [1, \ldots, N], \tilde{a}_i \in A_i :$$
$$\mathbb{E}_{\boldsymbol{a} \sim \boldsymbol{x}}[u_i(\boldsymbol{a})] + \varepsilon_i \geq \mathbb{E}_{\boldsymbol{a}_{-i} \sim \boldsymbol{x}_{-i}}[u_i(\tilde{a}_i, \boldsymbol{a}_{-i})]\},$$

where $\varepsilon_i = \max\{\frac{\|u_i\|_\infty}{1-\gamma_m}, \gamma_s, 2\gamma_p\}$. Hence, we can train policies robust to such agents by using the following in the robustness objective of Equation 1:

$$\boldsymbol{X}_\varepsilon(x_0) = \{\boldsymbol{x} \in P(A) \mid \exists x_0 \in A_0 : (x_0, \boldsymbol{x}) \in \text{CCE}_\varepsilon\}.$$

This proposition motivates us to use approximate CCE as an uncertainty set rather than exact CCEs. Conveniently, the algorithm we construct in the proof of Theorem 1 can be modified to adversarially sample from ϵ-CCE instead of exact CCE. By relaxation of Lemma 1, it yields the same optimality and runtime guarantees as Theorem 1, but over the

set of approximate ϵ-CCE. We will refer to this modification of the Theorem 1 algorithm as Algorithm 2, which we repeat in full in the Appendix.

5 Finding Approximate Uncertainty Sets with Blackbox Optimizers

One challenge with using Algorithm 2 in practice is that it relies on a no-regret learning subprocedure that does not scale well. This is a common bottleneck in multi-agent learning when deriving practical algorithms from algorithms with strong theoretical guarantees. A common remedy is to replace the no-regret learning procedure with a standard learning algorithm (e.g., SGD), usually with no negative impact on empirical behavior or performance. We will do exactly this to derive a practical variant of Algorithm 2 that still inherits its theoretical intuitions. This involves two main steps.

Removing binary search. Algorithm 2's binary search over y is a theoretically efficient search over possible optimal or worst-case values of ϵ-CCE. However, it is inefficient in a nested optimization like Equation 1. Observing that y's value affects only the parameterization of the importance weight β_{m+1}, we can fix v_{m+1} to a sufficiently small value such that $\beta_{m+1} = 1[v_1, \ldots, v_m \leq 0]$.

Replacing Blackwell's algorithm. During Algorithm 2's reduction to halfspace oracles, we can merge components of v corresponding to the same agent. This yields a functional equivalent of Eq 3,

$$v(x) = \begin{bmatrix} \max_{a_1 \in A_1} \overline{u}_1(a_1, \boldsymbol{x}_{-1}) - \overline{u}_1(\boldsymbol{x}) \\ \vdots \\ \max_{a_n \in A_n} \overline{u}_n(a_n, \boldsymbol{x}_{-n}) - \overline{u}_n(\boldsymbol{x}) \\ \sigma \to 0 \end{bmatrix}. \quad (4)$$

In practice, this choice is more tractable than Eq 3. Eq 4 lends itself to many efficient approximations. For instance, when A is combinatorially large, we can approximate the regret estimates with local methods rather than explicitly enumerating all possible action deviations.

A practical algorithm. Combined, these two modifications to Algorithm 2 render it equivalent to computing an upper-bound for the Lagrangian dual problem, $L(\epsilon)$, of adversarial sampling (Equation 7),

$$L(\epsilon) = \min_{\boldsymbol{x} \in P(A)} \max_{\lambda} \overline{u}_0(\boldsymbol{x}) - \sum_{i=1}^{N} \lambda_i \left[\mathrm{Reg}_i(\boldsymbol{x}) - \epsilon \right], \quad (5)$$

$$\mathrm{Reg}_i(\boldsymbol{x}) := \max_{a_i \in A_i} \overline{u}_i(a_i, \boldsymbol{x}_{-i}) - \overline{u}_i(\boldsymbol{x}).$$

Here, decoupled no-regret dynamics efficiently approximate the outer optimization $\min_{\boldsymbol{x} \in P(A)}$. In this sense, the theoretical results of Section 4 can be interpreted as a formal bound on how much decoupled approximations of $\min_{\boldsymbol{x} \in P(A)}$ affect the lower-bound of this dual problem. Theorem 1 can thus be interpreted as the implication that, when playing a sufficiently smooth game, it is reasonable to use decoupled algorithms to approximate the dual problem. This motivates our final modification to Algorithm 2: replacing the inner no-regret learning loop with a blackbox self-play algorithm. The final algorithm is described in Algorithm 1.

Algorithm 1: Decoupled sampling of pessimistic equilibria.

> **Output:** Approximate lower-bound on $L(\epsilon)$ (Eq 5).
> **Input:** Number of training steps M_{tr} and self-play steps M_s, reward slack ϵ, multiplier learning rate α_λ, uncoupled self-play algorithm B, regret estimators $R_i : P(A) \to \mathbb{R}$ for each agent i.
> Initialize mixed strategy $\boldsymbol{x}_1$.
> **for** $j = 1, \ldots, M_{\mathrm{tr}}$ **do**
>   **for** $i = 1, \ldots, N$ **do**
>   Estimate regret r_i as $\hat{r}_i \leftarrow R_i(\boldsymbol{x}_j)$, where
> $r_i := \max_{\tilde{x}_i \in P(A_i)} \overline{u}_i(\tilde{x}_i, \boldsymbol{x}_{-i}) - \overline{u}_i(\boldsymbol{x})$.
>   Compute multiplier $\lambda_i \leftarrow \lambda_i - \alpha_\lambda(\hat{r}_i - \epsilon)$.
>   **end for**
>   Using B, run M_s rounds of self-play with utilities $\hat{u}_i(\boldsymbol{a}) := (\lambda_i u_i(\boldsymbol{a}) - u_0(\boldsymbol{a}))/(1 + \lambda_i)$.
>   Set $\boldsymbol{x}_{j+1}$ as the resulting empirical play distribution.
> **end for**
> Return $\frac{1}{M_{\mathrm{tr}}} \sum_{t=1}^{M_{\mathrm{tr}}} \overline{u}_0(\boldsymbol{x}_t)$.

Adversarial Sampling Experiments

Before applying Algorithm 1 to more ambitious mechanism design tasks, we first benchmark the quality of its adversarial sampling. As our eventual mechanism design applications are spatiotemporal games requiring multi-agent reinforcement learning (MARL), for this experiment, we will also use MARL as the blackbox self-play procedure of Algorithm 1. In particular, we will use a common multi-agent implementation of the PPO algorithm (Schulman et al. 2017) and a Monte-Carlo sampling scheme as our regret estimator.

Game Environment. The game environment for this experiment is a Sequential Bimatrix Game. This is an extension of the classic *repeated bimatrix game* (Figure 2), whose Nash equilibria can be solved efficiently and is well-studied in game theory. At each timestep t, a row (agent 1) and column player (agent 2) choose how to move around a 4×4 grid, while receiving rewards $r_1(s_i, s_j), r_2(s_i, s_j)$. The current location is at row s_i and column s_j. The row (column) player chooses whether to move up (left) or down (right). Each episode is 500 timesteps.

We configure the payoff matrices r_1 and r_2, illustrated in Figure 2, so that only one Nash equilibrium exists and that the equilibrium constitutes a "tragedy-of-the-commons," where agents selfishly optimizing their own reward leads to less reward overall. The principal is a passive observer that observes the game and receives a payoff $r_0(s_i, s_j)$. The principal does not take any actions and its payoff is constructed such that its reward is high when the agents are at the Nash equilibrium. If Algorithm 1 successfully samples realistic worst-case behaviors, we expect to see agents 1 and 2 learning to deviate from their tragedy-of-the-commons equilibrium in order to (1) reduce the principal's reward but also (2) without significantly increasing their own regret.

Algorithm 1 efficiently interpolates between adversarial and low-regret equilibria. In Figure 2 (middle), we see
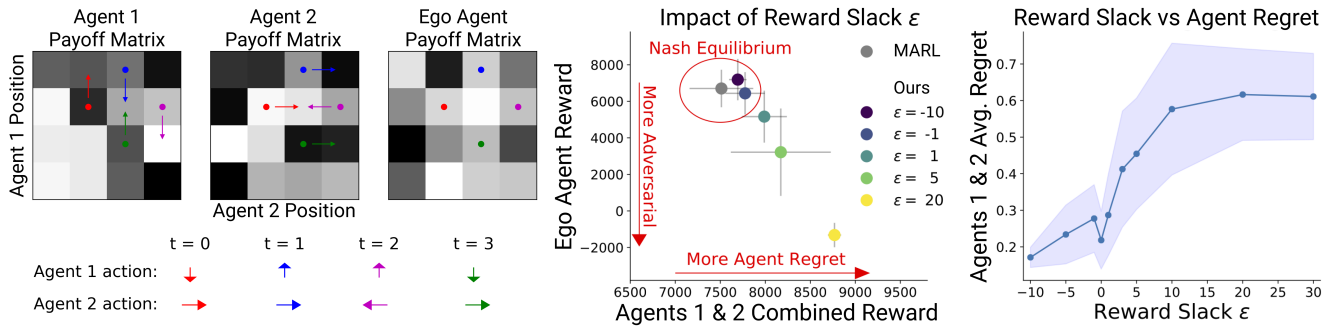
Figure 2: Validating our algorithm (Algorithm 3) in constrained repeated bimatrix games. **Left:** In the repeated bimatrix game, two agents navigate a 2D landscape. Both agents and the principal receive rewards based on visited coordinates. Brighter squares indicate higher payoff. The bimatrix reward structure encodes a social dilemma featuring a Nash equilibrium with low reward for the two agents and high reward for the gambler. Vanilla MARL converges to this equilibrium. **Right:** Agents trained with our algorithm deviate from this equilibrium in order to reduce the reward of the principal. ϵ governs the extent of the allowable deviation. As ϵ increases, the average per-timestep regret experienced by the agents also increases. Each average is taken over the final 12 episodes after rewards have converged. Each point in the above scatter plots describes the average outcome at the end of training for the agents (x-coordinate) and the principal (y-coordinate). Error bars indicate standard deviation.

the equilibria reached by agents balance the reward of the principal (y-axis) and themselves (x-axis). In particular, we see that conventional multi-agent RL discovers the attractive Nash equilibrium, which is in the top left. At this equilibrium, the agents do not cooperate and the principal receives high reward. Similarly, for small values of ϵ, our algorithm discovers the Nash equilibrium. Because ϵ acts as a constraint on agent regret, with larger values of ϵ, our algorithm deviates farther from the Nash equilibrium, discovering ϵ-equilibria to the bottom-right that result in lower principal rewards.

Algorithm 1 has tight control over how much agents sacrifice to hurt the principal. We see in Figure 2 (right) that deviations from the Nash equilibrium yield higher regret for the agents, i.e., regret increases with ϵ. This figure also confirms that increasing our algorithm's slack parameter correctly increases the incentive of the agents to incur regret in order to harm the principal.

6 Optimizing Strategic Robustness

We now apply our adversarial sampling scheme, Algorithm 1, to automated mechanism design problems. In particular, we will use Algorithm 1 to provide feedback to a reinforcement learning (RL) procedure that selects mechanisms. This RL procedure, Algorithm 3, is described in the Appendix for completeness. First, we induce a mechanism design problem on a repeated n-matrix game. Then, we'll seek to learn an optimal tax policy in the AI Economist, a large-scale spatiotemporal simulation of an economy (Zheng et al. 2020). Each experiment setting features 4 to 5 agents involved in complex multi-timestep interactions. They are thus significantly more complex and costly to train in than traditional multi-agent RL environments.

Repeated Matrix Games.

We first extend our repeated bimatrix game to include additional players and a principal.

Setup. The setting is now a 4-player, general-sum, normal-form game on a randomly generated $7 \times 7 \times 7 \times 7$ payoff matrix, with the same action and payoff rules as shown in Figure 2. Recall that these players will engage one another for 500 timesteps in an episode. Each player i is associated with an "original" reward function r_i; this is the reward function we will have access to during training. During test time, we may also encounter other categories of agents that have different reward functions but who are also themselves learned with RL.

1. *Vanilla*: r_i.
2. *Adversarial* (Adv): $r'_i = r_i - Qr_0$; larger Q is more adversarial.
3. *Risk-averse* (RiskAv): $r'_i = (r_i^{1-\eta} - 1)/(1 - \eta)$; higher η is more risk averse.

Results. Figure 3 shows the average rewards of principals trained (rows) and evaluated (columns) on each type of agents. We observe three key trends. First, principals trained on one type of agent generally perform better when evaluated on the same type of agent (diagonal entries). Second, principals trained with our adversarial sampling scheme perform better across the board. Third, the robustness gains of our adversarial sampling scheme are stronger when ϵ is large. This is expected as ϵ parameterizes the adversarial strength of our sampling scheme. For small ϵ, adversarial sampling reduces to random sampling as the CCE constraint is so tight it permits no adversarial deviations. We also saw that, even though all methods were run with 20 seeds and filtered down to 10 seeds on a validation set, our algorithm's results remain somewhat noisy, as it may not converge when badly initialized.

Taxing a Simulated Economy.

We now apply our algorithm to designing dynamic (multi-timestep) tax policies for a simulated trade-and-barter econ-

Training ↓ Testing →	Original	Adv ($Q=0.25$)	Adv ($Q=1$)	RiskAv ($\eta=0.05$)	RiskAv ($\eta=0.2$)
MARL	$104_{\pm50.5}$	$-5.8_{\pm34.1}$	$-232_{\pm29.3}$	$383_{\pm14.4}$	$352_{\pm42.6}$
Adv ($Q=0.25$)	$\mathbf{143}_{\pm35.0}$	$\mathbf{64.7}_{\pm35.1}$	$-191_{\pm36.0}$	$236_{\pm23.8}$	$292_{\pm28.4}$
Adv ($Q=1$)	$131_{\pm63.1}$	$-23_{\pm10.1}$	$\mathbf{-47}_{\pm8.20}$	$286_{\pm35.1}$	$290_{\pm36.0}$
RiskAv ($\eta=0.05$)	$-20_{\pm44.2}$	$-112_{\pm15.7}$	$-222_{\pm29.3}$	$\mathbf{404}_{\pm47.2}$	$\mathbf{464}_{\pm0.95}$
RiskAv ($\eta=0.2$)	$-53_{\pm32.4}$	$-150_{\pm20.0}$	$-283_{\pm29.8}$	$\mathbf{465}_{\pm0.61}$	$358_{\pm70.9}$
Ours $\epsilon=-10$	$\mathbf{227}_{\pm50.8}$	$\mathbf{48.9}_{\pm12.4}$	$-137_{\pm43.1}$	$265_{\pm41.8}$	$292_{\pm38.3}$
Ours $\epsilon=50$	$\mathbf{221}_{\pm80.8}$	$48.3_{\pm29.7}$	$\mathbf{-53}_{\pm14.0}$	$\mathbf{460}_{\pm33.9}$	$\mathbf{481}_{\pm38.6}$

Figure 3: **Robust performance in N-agent matrix games.** We train an principal in a $7 \times 7 \times 7 \times 7$ matrix game with $n=4$ agents (including the principal) until convergence. For each method, we train 20 seeds and select the top 10 in a validation environment. Each row corresponds to a specific agent type that the principal is trained on. 'MARL' refers to agents trained using their 'Original' reward definition; 'Adv' refers to adversarial agents; 'RiskAv' refers to risk-averse agents. The principals trained on these types of agents tend to perform best when evaluated on the same type seen during training. In contrast, principals trained against agent behaviors sampled using our algorithm ($\epsilon=50$) perform within standard error of top-1 on all agent types. We use the 'Original' reward definition when training with our algorithm.

Training ↓ Testing →	Original	$\eta=0.11$	$\eta=0.19$	$\eta=0.27$	$\alpha=0.25$	$\alpha=2.5$
Free Market	$326_{\pm1}$	$527_{\pm2}$	$427_{\pm1}$	$\mathbf{162}_{\pm1}$	$248_{\pm2}$	$112_{\pm0}$
Federal	$335_{\pm8}$	$637_{\pm5}$	$497_{\pm2}$	$150_{\pm2}$	$\mathbf{270}_{\pm1}$	$121_{\pm0}$
Saez	$\mathbf{381}_{\pm1}$	$597_{\pm3}$	$487_{\pm4}$	$\mathbf{189}_{\pm1}$	$265_{\pm0}$	$127_{\pm0}$
Ours ($\epsilon=-30$)	$\mathbf{375}_{\pm9}$	$\mathbf{646}_{\pm6}$	$\mathbf{514}_{\pm12}$	$164_{\pm9}$	$266_{\pm2}$	$131_{\pm2}$
AI Economist (Original)	$\mathbf{386}_{\pm2}$	$628_{\pm5}$	$515_{\pm1}$	$123_{\pm13}$	$267_{\pm0}$	$129_{\pm1}$
AI Economist ($\eta=0.11$)	$253_{\pm5}$	$\mathbf{683}_{\pm7}$	$506_{\pm1}$	$140_{\pm1}$	$255_{\pm2}$	$129_{\pm0}$
AI Economist ($\eta=0.19$)	$308_{\pm17}$	$\mathbf{665}_{\pm9}$	$\mathbf{543}_{\pm6}$	$82_{\pm29}$	$256_{\pm3}$	$131_{\pm1}$
AI Economist ($\eta=0.27$)	$339_{\pm11}$	$603_{\pm3}$	$477_{\pm1}$	$137_{\pm10}$	$266_{\pm0}$	$129_{\pm0}$
AI Economist ($\alpha=0.25$)	$324_{\pm2}$	$625_{\pm10}$	$501_{\pm7}$	$121_{\pm25}$	$263_{\pm0}$	$128_{\pm0}$
AI Economist ($\alpha=2.5$)	$104_{\pm27}$	$636_{\pm3}$	$246_{\pm33}$	$49_{\pm10}$	$251_{\pm4}$	$\mathbf{135}_{\pm1}$

Figure 4: **Robust dynamic tax policies in a spatiotemporal economy**. First 3 rows are classic tax baselines: "Free Market" has no taxes; "US Federal" uses the 2018 US Federal progressive income tax rates; "Saez" uses an adaptive, theoretical formula to estimate optimal tax rates. Bottom rows correspond to learned policies trained to optimize, and evaluated on, the social welfare metric *swf* of equality and productivity. 'Ours' and 'AI Economist (Original)' are trained on the 'Original' settings (risk aversion $\eta=0.23$; entropy bonus $\alpha=0.025$). Naive multi-agent reinforcement learning tax policies, including (Zheng et al. 2020)'s original AI Economist, fail to generalize to previously unseen agent types. In contrast, our algorithm performs within standard error of top-1 on all agent types.

omy with strategic taxpayers that interact with one another (Zheng et al. 2020). See Appendix B for a screenshot.

Setup. In this simulated economy, the principal sets a tax policy and the agents play a partially observable game, given the tax policy. Each episode is 1,000 timesteps of economic activity. Taxpayers earn income $z_{i,t}$ from labor $l_{i,t}$ and pay taxes $T(z_{i,t})$. They optimize their expected isoelastic utility:

$$\tilde{z}_{i,t} = z_{i,t} - T(z_{i,t}), \quad r_{i,t}(\tilde{x}_{i,t}, l_{i,t}) = \frac{\tilde{x}_{i,t}^{1-\eta} - 1}{1 - \eta} - l_{i,t}, \tag{6}$$

where $\tilde{x}_{i,t}$ is the post-tax endowment of agent i, and $\eta > 0$ sets the degree of risk aversion (higher η means higher risk aversion) (Arrow 1971). Players expend labor and earn income by participating in a rich simulated grid-world with resources and markets. The principal optimizes for social welfare $swf = \left(1 - \frac{N}{N-1}gini(z)\right) \cdot \left(\sum_{i=1}^{N} z_i\right)$, the product of equality (Gini 1912) and productivity. The taxpayers are also themselves learned with multi-agent RL, using a PPO algorithm entropy hyperparameter α (Schulman et al. 2017).

Results. Figure 4 shows the social welfare achieved by our algorithm, naive dynamic RL policies (Zheng et al. 2020), and static baseline tax policies (Saez, US Federal). Naive RL policies achieve good test performance when evaluated on the same agents seen in training, but perform poorly with agents with different η and noise level. They are often outperformed by the baseline taxes, which perform surprisingly well under strong risk aversion ($\eta = 0.27$) and noisy agents (entropy bonus $\alpha = 0.25, 2.5$). We see that the static baseline taxes may be more robust than dynamic ones, even in complex environments. However, our algorithm closes this robustness gap, consistently outperforming or tying both AI Economists and baseline taxes.

7 Future Work

Efficient sampling of worst-case equilibria is a key challenge for robust decision-making, and by extension, automated mechanism design. As we've explored uncertainty sets based on game-theoretic concepts, future work may build uncertainty sets that use domain knowledge or historical data and that may yield robustness to other types of domain shifts, e.g., in game dynamics.

References

Aghassi, M.; and Bertsimas, D. 2006. Robust game theory. *Mathematical Programming*, 107(1): 231–273.

Arrow, K. J. 1971. The theory of risk aversion. *Essays in the theory of risk-bearing*, 90–120. Publisher: Markham Chicago.

Barman, S.; and Ligett, K. 2015. Finding Any Nontrivial Coarse Correlated Equilibrium Is Hard. *ACM SIGecom Exchanges*, 14.

Basilico, N.; Coniglio, S.; Gatti, N.; and Marchesi, A. 2020. Bilevel programming methods for computing single-leader-multi-follower equilibria in normal-form and polymatrix games. *EURO Journal on Computational Optimization*, 8(1): 3–31.

Bonabeau, E. 2002. Agent-based modeling: Methods and techniques for simulating human systems. *Proceedings of the National Academy of Sciences*, 99(suppl 3): 7280–7287.

Cesa-Bianchi, N.; and Lugosi, G. 2006. *Prediction, Learning, and Games*. ISBN 978-0-521-84108-5. Journal Abbreviation: Prediction, Learning, and Games Publication Title: Prediction, Learning, and Games.

Conitzer, V.; and Sandholm, T. 2006. Computing the optimal strategy to commit to. In *Proceedings of the 7th ACM conference on Electronic commerce - EC '06*, 82–90. Ann Arbor, Michigan, USA: ACM Press. ISBN 978-1-59593-236-5.

Daskalakis, C.; Goldberg, P. W.; and Papadimitriou, C. H. 2009. The Complexity of Computing a Nash Equilibrium. *SIAM Journal on Computing*, 39(1): 195–259. Publisher: Society for Industrial and Applied Mathematics.

Duetting, P.; Feng, Z.; Narasimhan, H.; Parkes, D. C.; and Ravindranath, S. S. 2019. Optimal Auctions through Deep Learning. In Chaudhuri, K.; and Salakhutdinov, R., eds., *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, 1706–1715. PMLR.

Foster, D. P.; and Vohra, R. V. 1997. Calibrated Learning and Correlated Equilibrium. *Games and Economic Behavior*, 21(1): 40–55.

Fudenberg, D.; and Levine, D. K. 1989. Reputation and Equilibrium Selection in Games with a Patient Player. *Econometrica*, 57(4): 759–778. Publisher: [Wiley, Econometric Society].

Gini, C. 1912. *Variabilità e mutabilità*.

Gray, J.; Lerer, A.; Bakhtin, A.; and Brown, N. 2021. Human-Level Performance in No-Press Diplomacy via Equilibrium Search. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Hart, S.; and Mas-Colell, A. 2000. A Simple Adaptive Procedure Leading to Correlated Equilibrium. *Econometrica*, 68(5): 1127–1150.

Hazan, E.; and Koren, T. 2016. The computational power of optimization in online learning. In Wichs, D.; and Mansour, Y., eds., *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, 128–141. ACM.

Holland, J. H.; and Miller, J. H. 1991. Artificial Adaptive Agents in Economic Theory. *American Economic Review*, 81(2): 365–71.

Hou, L.; Pang, L.; Hong, X.; Lan, Y.; Ma, Z.; and Yin, D. 2020. Robust Reinforcement Learning with Wasserstein Constraint. *arXiv:2006.00945 [cs, stat]*.

Jiang, A. X.; and Leyton-Brown, K. 2011. A General Framework for Computing Optimal Correlated Equilibria in Compact Games. In Chen, N.; Elkind, E.; and Koutsoupias, E., eds., *Internet and Network Economics*, Lecture Notes in Computer Science, 218–229. Berlin, Heidelberg: Springer. ISBN 978-3-642-25510-6.

Kiekintveld, C.; Islam, T.; and Kreinovich, V. 2013. Security games with interval uncertainty. In *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, AAMAS '13, 231–238. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems. ISBN 978-1-4503-1993-5.

Korzhyk, D.; Conitzer, V.; and Parr, R. 2011. Security Games with Multiple Attacker Resources. In Walsh, T., ed., *IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July 16-22, 2011*, 273–279. IJCAI/AAAI.

Kreps, D. M.; and Wilson, R. 1982. Reputation and imperfect information. *Journal of Economic Theory*, 27(2): 253–279.

Li, S.; Wu, Y.; Cui, X.; Dong, H.; Fang, F.; and Russell, S. J. 2019. Robust Multi-Agent Reinforcement Learning via Minimax Deep Deterministic Policy Gradient. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, 4213–4220. AAAI Press.

Littlestone, N. 1987. Learning Quickly When Irrelevant Attributes Abound: A New Linear-threshold Algorithm. *Mach. Learn.*, 2(4): 285–318.

Morimoto, J.; and Doya, K. 2000. Robust Reinforcement Learning. In Leen, T. K.; Dietterich, T. G.; and Tresp, V., eds., *Advances in Neural Information Processing Systems 13, Papers from Neural Information Processing Systems (NIPS) 2000, Denver, CO, USA*, 1061–1067. MIT Press.

Myerson, R. B. 2016. Mechanism Design. In *The New Palgrave Dictionary of Economics*, 1–13. London: Palgrave Macmillan UK. ISBN 978-1-349-95121-5.

Nguyen, T. H.; Sinha, A.; Gholami, S.; Plumptre, A.; Joppa, L.; Tambe, M.; Driciru, M.; Wanyama, F.; Rwetsiba, A.; and Critchlow, R. 2016. CAPTURE: A new predictive anti-poaching tool for wildlife protection. *Proceedings of 15th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*.

Nguyen, T. H.; Yang, R.; Azaria, A.; Kraus, S.; and Tambe, M. 2013. Analyzing the Effectiveness of Adversary Modeling in Security Games. In desJardins, M.; and Littman, M. L., eds., *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence, July 14-18, 2013, Bellevue, Washington, USA*. AAAI Press.

Papadimitriou, C. H.; and Roughgarden, T. 2008. Computing correlated equilibria in multi-player games. *Journal of the ACM*, 55(3): 14:1–14:29.

Pinto, L.; Davidson, J.; Sukthankar, R.; and Gupta, A. 2017. Robust Adversarial Reinforcement Learning. In Precup, D.; and Teh, Y. W., eds., *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, 2817–2826. PMLR.

Pita, J.; Jain, M.; Ordóñez, F.; Portway, C.; Tambe, M.; Western, C.; Paruchuri, P.; and Kraus, S. 2009. Using Game Theory for Los Angeles Airport Security. *AI Magazine*, 30: 43–57.

Pita, J.; Jain, M.; Tambe, M.; Ordóñez, F.; and Kraus, S. 2010. Robust solutions to Stackelberg games: Addressing bounded rationality and limited observations in human cognition. *Artificial Intelligence*, 174(15): 1142–1171.

Pita, J.; John, R.; Maheswaran, R.; Tambe, M.; and Kraus, S. 2012. A robust approach to addressing human adversaries in security games. In *ECAI 2012*, 660–665. IOS Press.

Roughgarden, T. 2015. Intrinsic Robustness of the Price of Anarchy. *Journal of the ACM*, 62(5): 32:1–32:42.

Schäfer, F.; Zheng, H.; and Anandkumar, A. 2020. Implicit competitive regularization in GANs. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, 8533–8544. PMLR.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal Policy Optimization Algorithms. *arXiv:1707.06347 [cs]*. ArXiv: 1707.06347.

Simon, H. A. 1976. From substantive to procedural rationality. In Kastelein, T. J.; Kuipers, S. K.; Nijenhuis, W. A.; and Wagenaar, G. R., eds., *25 Years of Economic Theory: Retrospect and prospect*, 65–86. Boston, MA: Springer US. ISBN 978-1-4613-4367-7.

Tessler, C.; Efroni, Y.; and Mannor, S. 2019. Action Robust Reinforcement Learning and Applications in Continuous Control. In Chaudhuri, K.; and Salakhutdinov, R., eds., *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, 6215–6224. PMLR.

Zheng, S.; Trott, A.; Srinivasa, S.; Naik, N.; Gruesbeck, M.; Parkes, D. C.; and Socher, R. 2020. The AI Economist: Improving Equality and Productivity with AI-Driven Tax Policies. *arXiv:2004.13332 [cs, econ, q-fin, stat]*.

A Appendix: Proofs

Proof of Lemma 1

First, we observe that finding a ν-maximizing CCE is equivalent to solving the linear program below. Writing the joint distribution $\boldsymbol{x}$ as a $\prod_{i=1}^{N} m_i$-size vector of joint densities, a ν-maximizing CCE is given by:

$$\max_{\boldsymbol{x}\in P(A)} \overline{\nu}(\boldsymbol{x}) \text{ s.t. } \forall i \in [1,\ldots,N], \forall a_i \in A_i : \tag{7}$$

$$\sum_{\boldsymbol{a}'\in A} \left[u_i(a_i, \boldsymbol{a}'_{-i}) - u_i(\boldsymbol{a}') \right] \boldsymbol{x}(\boldsymbol{a}') \leq 0.$$

Although the number of constraints in this LP is polynomial, the number of variables is non-trivially exponential. In addition, the polynomial expectation property only guarantees polynomial time expected utility computations for product distributions.

Lemma 1. Fix a $0 < y \leq \max_{\boldsymbol{x}\in\text{CCE}} \overline{\nu}(\boldsymbol{x})$. Assume there is a $\text{Poly}(1/\epsilon, n, m)$-time halfspace oracle that, given a vector $\beta \in \mathbb{R}^{1+m}_+$ with non-negative components, returns an $\boldsymbol{x} \in P(A)$ such that $\beta v(\boldsymbol{x}) \leq 0$, where

$$v(\boldsymbol{x}) = \begin{bmatrix} \overline{u}_1(1, \boldsymbol{x}_{-1}) - \overline{u}_1(\boldsymbol{x}) \\ \vdots \\ \overline{u}_1(m_1, \boldsymbol{x}_{-1}) - \overline{u}_1(\boldsymbol{x}) \\ \vdots \\ \overline{u}_N(1, \boldsymbol{x}_{-N}) - \overline{u}_N(\boldsymbol{x}) \\ \vdots \\ \overline{u}_N(m_N, \boldsymbol{x}_{-N}) - \overline{u}_N(\boldsymbol{x}) \\ y - \overline{\nu}(\boldsymbol{x}) \end{bmatrix} \tag{3}$$

Then, there is a $\text{Poly}(1/\epsilon, n, m)$-time algorithm that returns an ϵ-CCE $\boldsymbol{x}$ with $\overline{\nu}(\boldsymbol{x}) \geq y - \epsilon$.

Proof. We will prove our lemma constructively. This proof closely mirrors (Cesa-Bianchi and Lugosi 2006)'s proof of Blackwell approachability.

First, we establish approachability conditions. Note that $\boldsymbol{x}$ is an ϵ-CCE satisfying $\overline{\nu}(\boldsymbol{x}) \leq \epsilon + y$ if $\boldsymbol{x}$ satisfies $v(\boldsymbol{x}) \leq \epsilon$ component-wise; the first m components give that $\boldsymbol{x}$ is an ϵ-CCE and the last component gives that $\overline{\nu}(\boldsymbol{x}) \leq \epsilon + y$. Thus if $y \leq \max_{\boldsymbol{x}\in\text{CCE}} \overline{\nu}(\boldsymbol{x})$, then we know there exists a $\boldsymbol{x} \in P(A)$ such that $v(\boldsymbol{x}) \leq 0$. In other words, the closed convex set $v(\boldsymbol{x}) \leq 0$ is approachable.

For the remainder of this proof, we assume $y \leq \max_{\boldsymbol{x}\in\text{CCE}} \overline{\nu}(\boldsymbol{x})$ and prove that we will return a ϵ-CCE that satisfies $\overline{\nu}(\boldsymbol{x}) \leq \epsilon + y$. If y does not satisfy this condition, then we will not return a CCE $\boldsymbol{x}$ that satisfies $\overline{\nu}(\boldsymbol{x}) \leq y$ because no such $\boldsymbol{x}$ would exist. Let $\boldsymbol{x}^* := \text{argmax}_{\boldsymbol{x}\in\text{CCE}}\overline{\nu}(\boldsymbol{x})$.

Fix a choice of ϵ. Now, consider the following iterative algorithm where we index timesteps by $t = 1, 2, \ldots$ and produce a sequence of mixed strategies beginning with some arbitrary choice of $\boldsymbol{x}^{(0)} \in P(A)$. We now consider every timestep t where the average value of $v(\boldsymbol{x})$,

$$\overline{v}_t := \frac{1}{t-1} \sum_{i=1}^{t-1} v(\boldsymbol{x}^{(i)}),$$

does not satisfy $\overline{v}_t \leq \epsilon$ component-wise. If instead $\overline{v}_t \leq \epsilon$ holds component-wise, we can return $\overline{\boldsymbol{x}}$, a uniform distribution over $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(t-1)}$, and be done.

Otherwise, project $\overline{v}_t$ onto the negative orthant, $S := [-\infty, 0]^{1+m}$, to obtain a vector $\beta \in \mathbb{R}^{m+1}$,

$$a_t = \arg\min_{a\in S} \|\overline{v}_t - a\|$$

$$\beta_t = \frac{\overline{v}_t - a_t}{\|\overline{v}_t - a_t\|}.$$

Since S is closed and convex, β exists and is unique. Note that β must be non-negative in all components by definition of S. We then consult our halfspace oracle to return our next iterate $\boldsymbol{x}^{(t)} \in P(A)$ where $\beta_t v(\boldsymbol{x}^{(t)}) \leq 0$. A solution must exist as, for any t, $\boldsymbol{x}^*$ satisfies $\beta_t v(\boldsymbol{x}^*) \leq 0$.

We now prove this algorithm returns the desired output in $\text{Poly}(\frac{1}{\epsilon}, n, m)$ and hence $\text{Poly}(\frac{1}{\epsilon}, |I|)$ time.

Since $\overline{v}_{t+1} = \frac{t-1}{t}\overline{v}_t + \frac{1}{t}v(\boldsymbol{x}^{(t)})$, we may write,

$$\begin{aligned} d(\overline{v}_{t+1}, S)^2 &= \|\overline{v}_{t+1} - a_{t+1}\|^2 \\ &\leq \|\overline{v}_{t+1} - a_t\|^2 \\ &= \|\frac{t-1}{t}\overline{v}_t + \frac{1}{t}v(\boldsymbol{x}^{(t)}) - a_t\|^2 \\ &= \|\frac{t-1}{t}(\overline{v}_t - a_t) + \frac{1}{t}\left(v(\boldsymbol{x}^{(t)}) - a_t\right)\|^2 \\ &= \left(\frac{t-1}{t}\right)^2 \|\overline{v}_t - a_t\|^2 \\ &\quad + \left(\frac{1}{t}\right)^2 \|v(\boldsymbol{x}^{(t)}) - a_t\|^2 \\ &\quad + 2\frac{t-1}{t^2}(v(\boldsymbol{x}^{(t)}) - a_t) \cdot (\overline{v}_t - a_t) \end{aligned}$$

Let $u_{\max} := \max\{\|u_1\|_\infty, \ldots, \|u_N\|_\infty, \|\nu\|_\infty\}$ and $u_{\min} := -\max\{\|-u_1\|_\infty, \ldots, \|-u_N\|_\infty, \|-\nu\|_\infty\}$. Recall that by construction of G, $u_{\max}, u_{\min}$ is polynomial bounded by $|I|$ so without loss of generality, let us fix utilities to a unit ball: $u_{\max} \leq 1, u_{\min} \geq -1$. We can rearrange the equation to obtain,

$$\begin{aligned} t^2\|\overline{v}_{t+1} - a_{t+1}\|^2 &- (t-1)^2\|\overline{v}_t - a_t\|^2 \\ &\leq (m+1) + 2(t-1)(v(\boldsymbol{x}^{(t)}) - a_t) \cdot (\overline{v}_t - a_t) \end{aligned}$$

Then sum both sides of the inequality for $t = 1, \ldots, n$ with the left-hand side telescoping to become $n^2\|\overline{v}_{n+1} - a_{n+1}\|^2$. Dividing both sides by n^2,

$$\begin{aligned} \|\overline{v}_{n+1} &- a_{n+1}\|^2 \\ &\leq \frac{(m+1)}{n} + \frac{2}{n}\sum_{t=1}^{n}\frac{t-1}{n}(v(\boldsymbol{x}^{(t)}) - a_t) \cdot (\overline{v}_t - a_t) \\ &\leq \frac{(m+1)}{n} + \frac{2}{n}\sum_{t=1}^{n}\frac{t-1}{n}\beta_t(v(\boldsymbol{x}^{(t)}) - a_t) \cdot \|\overline{v}_t - a_t\| \end{aligned}$$

Because $\frac{t-1}{n}\|\overline{v}_t - a_t\| \in [0, 2], \beta_t, -a_t \geq 0$ component-wise,

and by construction of $\boldsymbol{x}_t$,

$$\|\overline{v}_{n+1} - a_{n+1}\|^2 \leq \frac{(m+1)}{n} + \frac{8}{n}\sum_{t=1}^{n}\beta_t(v(\boldsymbol{x}^{(t)}) - a_t)$$

$$\leq \frac{(m+1)}{n} + \frac{8}{n}\sum_{t=1}^{n}\beta_t(v(\boldsymbol{x}^{(t)}))$$

$$\leq \frac{(m+1)}{n}$$

By triangle inequality, after n steps, we have a $\frac{2}{\sqrt{n}}$-CCE $\boldsymbol{x}$ with $\overline{\nu}(\boldsymbol{x}) \geq y - \frac{2}{\sqrt{n}}$. Substituting $\epsilon = \frac{2}{\sqrt{n}}$, we have a Poly$(\frac{1}{\epsilon}, n, m)$ runtime concluding our proof of this lemma.

Suppose that $\boldsymbol{x}^t$ is chosen where we only have the guarantee that $(\beta_t)_{1:m}v(\boldsymbol{x}^t)_{1:m} \leq \epsilon_1$ and $(\beta_t)_{m+1}v(\boldsymbol{x}^t)_{m+1} \leq \epsilon_2$. Then, we would instead have $\|(\overline{v}_{n+1})_{1:m} - (a_{n+1})_{1:m}\|^2 \leq \frac{(m+1)}{n} + 8\epsilon_1$ and $\|(\overline{v}_{n+1})_{m+1} - (a_{n+1})_{m+1}\|^2 \leq \frac{(m+1)}{n} + 8\epsilon_2$. $\square$

Proof of Lemma 2

Lemma 2. Let ρ denote an upper bound on the price of anarchy of $G, \tilde{G}$. There exists a Poly$(1/\epsilon, n, m, \rho)$-time halfspace oracle that, given a vector $\beta \in \mathbb{R}_+^{1+m}$ and $y \leq \rho \max_{\boldsymbol{x} \in \text{CCE}} \overline{\nu}(\boldsymbol{x})$, returns an $\boldsymbol{x} \in P(A)$ such that $\beta v(\boldsymbol{x}) \leq \epsilon$ where v is defined as in Equation 3.

Proof. We will prove that this oracle exists by construction. We will further limit our choices of candidates $\boldsymbol{x}$ to product distributions. Then, finding our desired oracle is equivalent to solving an optimization problem over m variables: $x_1^1, \ldots, x_1^{m_1}, \ldots, x_N^1, \ldots, x_N^{m_N}$ where x_i^j corresponds to the probability that agent i plays action j under $\boldsymbol{x}$. The objective is minimizing

$$\beta v(\boldsymbol{x}) = -\beta_{m+1}(\overline{\nu}(\boldsymbol{x}) - y)$$

$$- \sum_{i=1}^{N}\left(\sum_{j=1}^{m_i}\beta_i^j\right)\left(\overline{u}_i(\boldsymbol{x}) - \frac{\sum_{j=1}^{m_i}\beta_i^j\sum_{\boldsymbol{a}_{-i}}x_{-i}^{\boldsymbol{a}_{-i}}u_i(j, \boldsymbol{a}_{-i})}{\sum_{j=1}^{m_i}\beta_i^j}\right)$$

$$= -\beta_{m+1}(\overline{\nu}(\boldsymbol{x}) - y)$$

$$- \sum_{i=1}^{N}\|\beta_i\|_1\left(\overline{u}_i(\boldsymbol{x}) - \overline{u}_i\left(\frac{\beta_i}{\|\beta_i\|_1}, \boldsymbol{x}_{-i}\right)\right)$$

down to a non-positive value. For convenience, we'll write,

$$C(\boldsymbol{x}) = -\beta_{m+1}\overline{v}(\boldsymbol{x})$$

$$- \sum_{i=1}^{N}\|\beta_i\|_1\left(\overline{u}_i(\boldsymbol{x}) - \overline{u}_i\left(\frac{\beta_i}{\|\beta_i\|_1}, \boldsymbol{x}_{-i}\right)\right)$$

Note that efficiently finding a $\boldsymbol{x}$ with a guaranteed negative upper bound lets us specify that upper bound as a condition y.

This optimization problem is combinatorial and generally intractable. However, we can exploit the usually high social welfare of no-regret learning. Consider no-regret learning with n agents that have the cost function,

$$c_i(\boldsymbol{x}) = -\frac{\beta_{m+1}\overline{\nu}(\boldsymbol{x})}{n} - \|\beta_i\|_1\left(\overline{u}_i(\boldsymbol{x}) - \overline{u}_i\left(\frac{\beta_i}{\|\beta_i\|_1}, \boldsymbol{x}_{-i}\right)\right),$$

with action space A_i. Since applying regret matching would ignore our $\overline{\nu}$ terms, we instead take advantage of smoothness. Observe that the first term (the game of penalizing principal agent) is bound by the smoothness of $\tilde{G}$, and the second term (the game of minimizing incentive to deviate from β) is bound by the smoothness of G. Thus, for any $\boldsymbol{x}, \boldsymbol{x}^*$:

$$\sum_{i=1}^{N}\frac{\overline{\nu}(\boldsymbol{x}_{-i}, x_i^*)_{m+1}}{N} \leq \sum_{i=1}^{N}-\lambda_{\tilde{G}}\frac{\overline{\nu}(\boldsymbol{x}^*)}{N} - \mu_{\tilde{G}}\frac{\overline{\nu}(\boldsymbol{x})}{N};$$

$$\sum_{i=1}^{N}\overline{\nu}(\boldsymbol{x}_{-i}, x_i^*)_{1:m} \leq \sum_{i=1}^{N}\|\beta_i\|\lambda_G\left(\overline{u}_i(\beta) - \overline{u}_i(\beta)\right)$$

$$+ \|\beta_i\|\mu_G\left(\overline{u}_i\left(\frac{\beta_i}{\|\beta_i\|_1}, \boldsymbol{x}_{-i}\right) - \overline{u}_i(\boldsymbol{x})\right).$$

We can now run a generic no-regret learning algorithm, say Hedge, which has regret $O(\sqrt{T})$ after T episodes. Specifically, the ergodic average $\boldsymbol{x}$ of T episodes is a $O(\frac{1}{\sqrt{T}})$-CCE. Letting $\boldsymbol{x}^*$ be a CCE for the game defined by $\{c_i\}_{i=1}^N$, it follows:

$$\beta v(\boldsymbol{x}) \leq -\beta_{m+1}(\lambda_{\tilde{G}}\overline{\nu}(\boldsymbol{x}^*) + \mu_{\tilde{G}}\overline{\nu}(\boldsymbol{x}))$$

$$+ \mu_G\beta_{1:m}v(\boldsymbol{x})_{1:m} + O(\sqrt{T}).$$

Finally, recall we require that $C(\boldsymbol{x}) + \beta_{m+1}y = \beta v(x) \leq 0$, meaning that in order for our halfspace oracle to always return something, we require y satisfy:

$$\frac{\lambda \max_{\boldsymbol{x} \in \text{CCE}}\overline{\nu}(\boldsymbol{x})}{\mu - 1} + \beta_{m+1}y \leq 0$$

$$\to y \leq \frac{\lambda \max_{\boldsymbol{x} \in \text{CCE}}\overline{\nu}(\boldsymbol{x})}{\beta_{m+1}(1-\mu)} \leq \frac{\lambda \max_{\boldsymbol{x} \in \text{CCE}}\overline{\nu}(\boldsymbol{x})}{(1-\mu)}$$

$\square$

Proof of Theorem 1

Theorem 1. For succinct n-agent m-action games of polynomial type and expectation property, there exists a Poly$(1/\epsilon, n, m, \rho)$ algorithm that will find an ϵ-CCE $\boldsymbol{x}$ with

$$\overline{\nu}(\boldsymbol{x}) \geq \frac{y}{\rho} - \epsilon.$$

for any $y \leq \max_{\boldsymbol{x}^* \in \text{CCE}}\overline{\nu}(\boldsymbol{x}^*)$, where $\rho = \frac{\lambda_{\tilde{G}}}{1-\max\{\mu_G, \mu_{\tilde{G}}\}}$.

The main theorem follows directly from Lemmas 1, 2. Specifically, for any $y \leq \max_{\boldsymbol{x} \in \text{MNE}}$, instantiate the algorithm described in Lemma 1 with the oracle described in Lemma 2. This is described in full in Algorithm 2. Note that the oracle returns a mixed strategy that is guaranteed to be composed of a polynomial number of known product distributions; hence, any mixed strategies returned by the oracle can be evaluated in polynomial time.

Proof of Proposition 1

We first motivate our study of how behavioral deviations affect the generalization of strategic policies. Specifically,

we will look at examples of where naive strategic decision-making can result in significant regret when confronted with boundedly rational agents—even when said agents are asymptotically rational.

For simplicity, we'll discuss a traditional single-follower Stackelberg setting where $N = 1$. In this setting, we'll refer to agent 0 as the leader and to agent 1 as the follower. Recall that in fully rational settings, we can anticipate the follower to play a best response to whatever mixed strategy the leader adopts: $\boldsymbol{X} = \mathrm{BR}(x_0) := \mathrm{argmax}_{a_1 \in A_1} \mathbb{E}_{a_0 \sim x_0}[u_1(a_0, a_1)]$.

Consider an instance of bounded rationality where the leader plays a strategy x_0 but the follower perceives the leader as having committed to the strategy $\hat{x}_0$. This may arise in partial observability settings or when the follower is limited to bandit feedback, for example. A classical statement of the fragility of Stackelberg equilibria is that even when the follower's error, $\|x_0 - \hat{x}_0\|$, is infinitesimally small, the follower's bounded rationality can cost the leader a constant value bounded away from zero.

Proposition 2. Fix a leader mixed strategy $x_0 \in P(A_0)$. There exists Stackelberg game ($N = 1$) with bounded payoffs where there is a sequence of mixed strategies $x_0^{(1)}, x_0^{(2)}, \cdots \in P(A_0)$ such that $\lim_{t \to \infty} \|x_0 - x_0^{(t)}\|_\infty = 0$ but for all t, $\min_{x_1 \in \mathrm{BR}(x_0)} u_0(x_0, x_1) - \max_{x_1 \in \mathrm{BR}(x_0^{(t)})} u_0(x_0, x_1)$ is positive and bounded away from zero.

These situations arise when the relationship between the leader and follower's utility functions is not smooth. Samuelson's game is an example of where this may result in non-robust optimal strategies independent of a Stackelberg assumption. Consider the below payoff matrix.

	L	R
T	100, 100	50, 99
B	99, 100	99, 99

There is a unique Nash equilibria at $a_0 = T, a_1 = L$. Accordingly, optimizing against the best response of the column-player would recommend a T pure strategy to the row-player. However, if the column-player is boundedly rational or otherwise acts according to a perturbed payoff matrix, a safer strategy for the row-player is to play the action B, which guarantees at least a payoff of 99. Optimizing against the worst-case ϵ-best response of the column-player, where $\epsilon \geq 1$, would recommend this robust B strategy to the row-player. Extending this insight into the $N > 1$ domain, we similarly want to optimize against worst-case equilibria to afford robustness against payoff structures such as Samuelson games.

Proposition 1. The uncertainty set $\boldsymbol{X}'$ of (any combination of) agents violating assumptions 1-3 with parameters $\gamma_m, \gamma_s, \gamma_p$ is contained in the set of ε-CCE:
$$\mathrm{CCE}_\varepsilon = \{\boldsymbol{x} \in P(A) \mid \forall i \in [1, \ldots, N], \tilde{a}_i \in A_i :$$
$$\mathbb{E}_{\boldsymbol{a} \sim \boldsymbol{x}}[u_i(\boldsymbol{a})] + \varepsilon_i \geq \mathbb{E}_{\boldsymbol{a}_{-i} \sim \boldsymbol{x}_{-i}}[u_i(\tilde{a}_i, \boldsymbol{a}_{-i})]\},$$
where $\varepsilon_i = \max\{\frac{\|u_i\|_\infty}{1 - \gamma_m}, \gamma_s, 2\gamma_p\}$. Hence, we can train policies robust to such agents by using the following in the robustness objective of Equation 1:
$$\boldsymbol{X}_\varepsilon(x_0) = \{\boldsymbol{x} \in P(A) \mid \exists x_0 \in A_0 : (x_0, \boldsymbol{x}) \in \mathrm{CCE}_\varepsilon\}.$$

The proof of this proposition follows immediately from the observation that all three uncertainties—subjective rationality, procedural rationality, and myopicness/commitment power—can be framed as uncertainty about agent utility functions.

Proof. First, subjective rationality is by definition an uncertainty about agent utility functions. Our parameterization of this uncertainty yields a natural uncertainty set over reward functions. Recall that we upper bound the infinity norm by γ_s, which is defined as the $\inf_\gamma$ s.t. $\|u_i - \hat{u}_i\|_\infty := \max_{\boldsymbol{a} \in A} |u_i(\boldsymbol{a}) - \hat{u}_i(\boldsymbol{a})| \leq \gamma$. We accordingly define the uncertainty set $\Xi_\varepsilon(u_i) = \{u_i : A \to \mathbb{R} \mid \|u_i - \hat{u}_i\|_\infty \leq \varepsilon\}$.

Similarly, we can define an uncertainty set around non-myopic agents and general uncertainty about our commitment power by upper-bounding possible perturbations to an agent's originally anticipated utility function. In particular, $|u_i - (u_i(\boldsymbol{x}) + \sum_{t=1}^\infty \gamma_m^t u_i(\boldsymbol{x}_t))| \leq \frac{\|u_i\|_\infty}{1 - \gamma_m} = \epsilon$. Hence, all non-myopic agents with $\gamma_m \leq 1 - \frac{\|u_i\|_\infty}{\epsilon}$ will be expressed in the uncertainty set $\Xi_\varepsilon(u_i)$.

Procedural rationality can also be framed in terms of $\Xi_\varepsilon(u_i)$. In particular, the ϵ_i-coarse correlated equilibria under u_i exactly coincides with the union of coarse correlated equilibria under reward perturbations of up to $\epsilon/2$. Formally, $\boldsymbol{x}$ is in an ϵ_i-coarse correlated equilibria under u_i if and only if,
$$\forall i \in [1, \ldots, N], x_i \in P(A_i) : \overline{u}_i(\boldsymbol{x}) + \epsilon_i \geq \overline{u}_i(x_i, \boldsymbol{x}_{-i})$$
$\boldsymbol{x}$ is in a coarse correlated equilibria under some $u'_i \in \Xi_\varepsilon(u_i)$ if and only if,
$$\exists u'_i \in \Xi_\varepsilon(u_i), \text{ s.t.}$$
$$\forall i \in [1, \ldots, N], x_i \in P(A_i) : \overline{u'}_i(\boldsymbol{x}) \geq \overline{u'}_i(x_i, \boldsymbol{x}_{-i}),$$
which is equivalent to the condition,
$$\forall i \in [1, \ldots, N], x_i \in P(A_i) :$$
$$\overline{u'}_i(\boldsymbol{x}) + \varepsilon \geq \overline{u'}_i(x_i, \boldsymbol{x}_{-i}) - \varepsilon,$$

Hence, the set of $\frac{\varepsilon}{2}$-equilibria behaviors, up to $\left(1 - \frac{\|u_i\|_\infty}{\varepsilon}\right)$-discount factor non-myopic agent behaviors, and up to ε reward perturbation-consistent behaviors are all contained in the set of ε-CCE. $\square$

B Appendix: Additional Experiments

Ablation Study: Dynamic vs Fixed Lagrange multipliers.

In our robust learning framework, Algorithm 1, the Lagrange multipliers $\boldsymbol{\lambda}$ play an important rule in moderating the self-play dynamics used to sample adversarial dynamics. Specifically, the multiplier λ_i for agent i balances agent i's incentive to improve its own reward with its obligation to act adversarially to the principal by minimizing the principal's reward. Recall that smaller values of λ yield more antagonistic agent objectives. As described in Algorithm 3, these multipliers $\boldsymbol{\lambda}$ are periodically updated using local Monte-Carlo estimates of regret. This raises the question of whether we can instead fix a constant non-zero value for the multipliers $\boldsymbol{\lambda}$ and still retain an effective adversarial equilibria sampler. This experiment

answers that question in the negative. Figure 5 visualizes the equilibria discovered with a fixed λ_0 in the same format as in Figure 2 (middle). This comparison shows that using a fixed λ_0 affects the equilibria discovered by our algorithm; the bottom right quadrant which contains the ϵ-equilibria discovered with the dynamic multipliers λ used by Algorithm 3 are not reached for any values of frozen multipliers λ_0. This demonstrates that, even in simple game settings, certain ϵ-equilibria are only reachable with dynamic λ and hence multiplier updates are necessary for proper behavior.
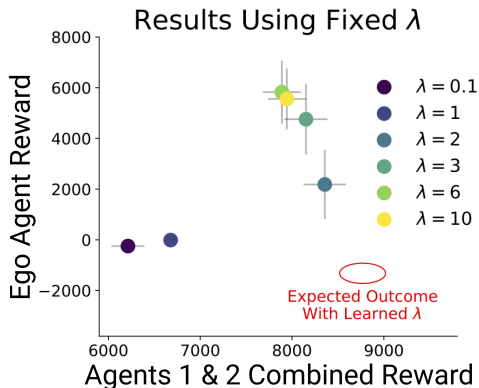

Figure 5: Using fixed values of λ (rather than allowing it to update, as in the full algorithm) distorts performance and prevents agents from reaching the same ϵ-equilibria discovered with learned λ.
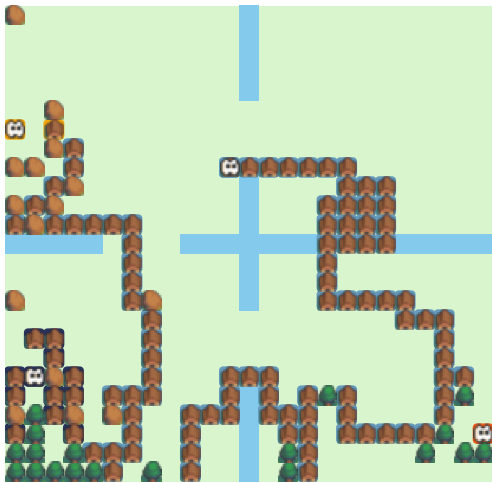

Figure 6: Visualization of the spatiotemporal economic simulation published by (Zheng et al. 2020) and used in our dynamic mechanism design experiment in Figure 4. In this 25-by-25 grid world, 4 heterogenous agents perform labor, trade resources, earn income, and pay taxes according to the schedules set by our policies.

Additional Experiment Details

The hyperparameters for our deep dynamic mechanism design experiments are listed in Table 1 for the N-Matrix games (depicted in Figure 3), and Table 2 (depicted in Figure 4) for the AI Economist tax policy game. Additional hyperparameters specific to the AI Economist simulation environment were kept at default values as described in the manuscript (Zheng et al. 2020).

Table 1 details: In the N-Matrix game experiments, we train 20 models for each training method and conduct evaluation runs of each model in every test environment. However, evaluation runs may fail to stabilize. Thus, we drop the bottom 10 evaluation runs for each training method.

Table 2 details: In the AI Economist tax policy experiments, we train 9 models for each training method and conduct evaluation runs of each model in every test environment. However, evaluation runs may fail to stabilize. Thus, we drop the bottom 6 evaluation runs for each training method. Furthermore, for this complex task, the adversary in Algorithm 3 may fail to converge in reasonable time. Thus, for the Algorithm 3 training method, we select its 9 trained models from a set of 15 candidate models (i.e., dropping at most 6 failed training runs). This selection is made programmatically on the basis of each model's median reward on validation runs in a standalone validation environment ($\eta = 0.27$). We similarly choose the $\epsilon = 30$ hyperparameter for Algorithm 3 from a grid-search over $\epsilon \in \{0, -10, -20, -30, -40\}$ on the basis of validation reward.

C   Appendix: Additional Related Work

In this section, we discuss additional related work regarding adversarially robust reinforcement learning. The topic of adversarially robust reinforcement learning originates from control theory literature. Morimoto and Doya (2000) proposed a form of adversarially robust reinforcement learning, inspired by $\mathcal{H}_\infty$ control, for attaining robustness to uncertainty about the environment dynamics. Later, Pinto et al. (2017) proposes to learn an adversary policy to induce this robustness, again targeting robustness to uncertainty about environment dynamics in a control-theoretic sense. Other works have since studied perturbing transition matrices, observation spaces, and action probabilities of Markov decision processes, with motivating applications in robotics and control theory (Tessler, Efroni, and Mannor 2019; Hou et al. 2020). These concepts have recently been extended to multi-agent settings, such as by (Li et al. 2019). However, these prior work on robust multi-agent learning differ from ours in two important ways. First, their multi-agent task is controlling a swarm (of multiple agents) in an efficient and coordinated fashion. Our multi-agent task is designing a policy that will yield favorable strategic outcomes in a multi-agent game. Second, their works analyze control-theoretic notions of robustness, extending single-agent policy techniques to a multi-agent policy setting.

Parameter	Value
Training algorithm	PPO
Episodes	10,000
CPUs (Baselines)	15
CPUs (Algorithm 3)	95
Environment parameters	
Number of agents	4
Episode length	500
World dimensions	7x7x7x7
Training Seeds	20
Test Seeds	10
Neural network parameters	
Number of convolutional layers	0
Number of fully-connected layers	2
Fully-connected layer dimension (agent)	128; 32
Fully-connected layer dimension (planner)	128; 32
LSTM cell size (agent)	0
LSTM cell size (planner)	0
All agents share weights	True
PPO Parameters	
Learning Rate (Principal)	0.0006
Learning Rate, η (Other Agents)	0.003
Entropy regularization (Principal)	0.1
Entropy regularization, α (Other Agents)	0.025
Gamma	0.998
GAE Lambda	0.98
Gradient clipping	10
Value function loss coefficient	0.05
SGD Minibatch Size	2500
SGD Sequence Length	50
Value/Policy networks share weights	False
Algorithm 3 Parameters	
n_{train}	4
n_{test}	10
Initial multipliers λ	8
Multiplier learning rate α_λ	0.01, 0.001

Table 1: Hyperparameters for N-Matrix Experiments

Parameter	Value
Training algorithm	PPO
Pretrained Episodes	450,000
Finetuned Episodes	20,000
CPUs (Baselines)	15
CPUs (Algorithm 3)	95
Environment parameters	
Number of agents	4
Episode length	1000
World dimensions	25x25
Default iso-elastic η	0.23
Training Seeds	9
Test Seeds	3
Neural network parameters	
Number of convolutional layers	2
Number of fully-connected layers	2
Fully-connected layer dimension (agent)	128
Fully-connected layer dimension (planner)	256
LSTM cell size (agent)	128
LSTM cell size (planner)	256
All agents share weights	True
PPO Parameters	
Learning Rate (Principal)	0.0001
Learning Rate, η (Other Agents)	0.0003
Entropy regularization (Principal)	0.1
Entropy regularization, α (Other Agents)	0.025
Gamma	0.998
GAE Lambda	0.98
Gradient clipping	10
Value function loss coefficient	0.05
SGD Minibatch Size	3000
SGD Sequence Length	50
Value/Policy networks share weights	False
Algorithm 3 Parameters	
n_{train}	2
n_{test}	5
Initial multipliers λ	20
Multiplier learning rate α_λ	0.01

Table 2: Hyperparameters for AI Economist Experiments

D   Appendix: Algorithm Pseudocodes

In this section, we provide the full pseudocode description of Algorithms 2 and 3.

Algorithm 2: Poly$(n, m, \rho, \varepsilon)$-time algorithm for sampling approximately optimal CCE.

Output: ϵ-CCE $\boldsymbol{x}$ such that

$$\overline{\nu}(x_0, \boldsymbol{x}) \geq \frac{1}{\rho} \max_{\tilde{\boldsymbol{x}} \in \epsilon\text{-CCE}} \overline{\nu}(x_0, \tilde{\boldsymbol{x}}) - \varepsilon.$$

Input: Generalization parameter $\epsilon \in \mathbb{R}_+^m$, tolerance ε, objective function $\nu : A_0 \times A \to \mathbb{R}$, principal policy x_0, bandit feedback access to utility functions $\overline{u}_0, \ldots, \overline{u}_N$.

while Binary search over $y \in \mathbb{R}$ (within ν's payoff bounds) is coarser than the desired Poly$(|I|)$ resolution **do**

Initialize some $\boldsymbol{x} \in P^{\text{prod}}(A)$.

for $O(\frac{1}{\varepsilon^2})$ iterations indexed by t **do**

Compute the vector v_t from Theorem 1 (Eq 3).

Compute importance weight vector,

$$\beta_t \leftarrow \begin{bmatrix} \max\{v_1 - \epsilon_1, 0\} \\ \vdots \\ \max\{v_{m_1} - \epsilon_1, 0\} \\ \vdots \\ \max\{v_{m-1} - \epsilon_N, 0\} \\ \vdots \\ \max\{v_m - \epsilon_N, 0\} \\ \max\{v_{m+1}, 0\} \end{bmatrix}$$

Normalize $\beta_t \leftarrow \beta_t / \|\beta_t\|_1$.

Run $O(\frac{1}{\varepsilon^2})$ steps of simultaneous deterministic no-regret dynamics on n agents with cost function,

$$c_i(\boldsymbol{x}) = -\frac{\beta_{m+1}\overline{\nu}(\boldsymbol{x})}{n}$$
$$- \|\beta_i\|_1 \left(\overline{u}_i(\boldsymbol{x}) - \overline{u}_i \left(\frac{\beta_i}{\|\beta_i\|_1}, \boldsymbol{x}_{-i} \right) \right),$$

where β_i are the components of β starting with index $1 + \sum_{j=1}^{i-1} m_j$ and ending with index $\sum_{j=1}^i m_j$ inclusive.

Store empirical play distribution of the no-regret dynamics as $\boldsymbol{x}_t$.

If $\beta_t \boldsymbol{x}_t \geq \varepsilon$, y is too large. Terminate loop and resume binary search.

end for

If the prior loop was successful (y was not too large), store the average empirical play distribution over $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots$ as $\boldsymbol{x}^*$.

end while

Return $\boldsymbol{x}^*$.

Recall that the primary difference between Algorithm 1 and Algorithm 2 is that Algorithm 2 invokes no-regret learning algorithms as a subprocedure, which are provably computationally inefficient (e.g., see Hazan and Koren (2016)). No-regret algorithms do not even exist for the online learn-ing of some finite VC dimension classes (Littlestone 1987), meaning there are no known implementations of Algorithm 2 for the experiments considered in the paper. In situations where one can reasonably expect gradient descent self-play to behave like a no-regret algorithm, Algorithm 1 can still be seen as an implementation of Algorithm 2.

Algorithm 3: Adversarially robust dynamic mecha-nism design.

Output: Learned parameters, θ_0, for a mechanism represented as agent 0.

Input: Reward slack ϵ, learning rate α_λ, batch size β, initial agent parameters $\boldsymbol{\theta} = [\theta_1, \ldots, \theta_N]$.

Input: Number of rounds n_{rounds}, evaluation batches n_{eval}, training batches n_{train}.

for $j = 1, \ldots, n_{\text{rounds}}$ **do**

Copy $\boldsymbol{\theta}$ into placeholders $\tilde{\boldsymbol{\theta}} \leftarrow \boldsymbol{\theta}$.

for $j = 1, \ldots, n_{\text{eval}}$ **do**

Accumulate β timesteps under $\tilde{\boldsymbol{\theta}}, \theta_0$.

For each agent $i \geq 1$, store their experience in a tuple of lists,

$$B_{i,j} := (\text{Rew}_{i,j}, \text{Obs}_{i,j}, \text{Actions}_{i,j}).$$

Update each agent $i \geq 1$ critic (if used) and actor for $\tilde{\boldsymbol{\theta}}$ with B_i.

end for

Update each agent $i \geq 1$ multiplier:

$$\lambda_i \leftarrow \lambda_i - \alpha_\lambda \text{Mean}\left(\text{Rew}_{i,n_{\text{rounds}}} - \epsilon - \text{Rew}_{i,0}\right)$$

for $j = 1, \ldots, n_{\text{train}}$ **do**

Accumulate β timesteps under $\boldsymbol{\theta}$ and θ_0 and store them in $B_{0,j}, \ldots, B_{N,j}$.

Update the mechanism's θ_0 critic and actor with B_0 using a slow learning rate.

Update each agent $i \geq 1$ critic for $\boldsymbol{\theta}$ (if used) with B_i.

Update each agent $i \geq$ actor for $\boldsymbol{\theta}$with,

$$\tilde{B}_{i,j} = \left(\tilde{\text{Rew}}_{i,j}, \text{Obs}_{i,j}, \text{Actions}_{i,j}\right)$$

where the recorded reward of agent i, at each timestep $t = 1, \ldots, \beta$, is modified:

$$\tilde{\text{Rew}}_{i,j} = \left[\frac{\lambda_i \text{Rew}_{i,j}^{(t)} - \text{Rew}_{0,j}^{(t)}}{1 + \lambda_i} \mid t = 1, \ldots, \beta \right]$$

end for
end for
Return parameters θ_0.