Towards Cooperative Learning Equilibrium in Reinforcement Learning

Jesse Clifton
Center on Long-Term Risk
jesse.clifton@longtermrisk.org

Maxime Riché
riche.maxime@altruismeeficacefrance.org

Abstract

We consider the problem of constructing learning equilibria in the multi-agent reinforcement learning setting. A learning equilibrium is a profile of learning algorithms which constitute a Nash equilibrium; that is, no player can benefit in terms of the long-term rewards generated by the profile of learning algorithms by deviating from that profile. We argue that learning equilibrium is a more appropriate solution concept for certain applications of multi-agent reinforcement learning than others which are frequently studied. We then propose an approach to constructing pairs of learning algorithms in which reinforcement learners jointly maximize a welfare function which represents some compromise between their individual payoffs. Deviations from the optimization of this welfare function are punished. We show that our construction is a learning equilibrium in the “perfect policy monitoring” case, where agents can verify whether their counterparts are cooperating. Then we present some proof-of-concept experiments in social dilemmas, for both the perfect and imperfect policy monitoring cases.

1 Introduction

As the capabilities of artificial agents improve, ensuring that these agents interact cooperatively is an increasingly important research direction. In the setting we envision, multiple reinforcement learners are deployed in an initially unknown environment to act on behalf of their respective principals. These principals will generally have differing goals, and so need to design their learners to avoid conflict while still advancing their own goals as well as possible.

A natural criterion for rational behavior on part of the principals is whether their learners are in learning equilibrium (LE) [Brafman and Tennenholtz, 2003]: do the learning algorithms constitute a Nash equilibrium, in terms of the expected cumulative rewards they generate? Beyond that, the learning equilibrium should be in some sense socially optimal. It should be Pareto-optimal, should reflect some combination of total utility gained by all of the agents, the fairness of the distribution of utility, and other normative considerations. Lastly, equilibrium must be enforced with punishment, but we want the punishment of deviations to be forgiving. In particular, punishments should only be severe enough to deter deviations from the equilibrium, be no more. Altogether, this can be thought of as a formal model of principals specifying a contract and a method for enforcing that contract.

The purpose of this work is to make progress towards these desiderata by sketching a new framework for multi-agent reinforcement learning. To measure the quality of different payoff profiles, we rely on a welfare function. The learning equilibrium we construct will converge to a policy profile that maximizes this welfare function. It is beyond the scope of this article to argue for a particular welfare function, as this is ultimately an ethical judgement to be made by the human principals. For the sake of the discussion here, we refer to several widely-used welfare functions from the social choice and cooperative bargaining literatures, which have some appealing theoretical properties (Table 1).

There are several benefits to this approach:

1. In this setting, LE is a better account of individual rationality than other solution concepts used in multi-agent learning. The principals will want to choose their learners strategically. This means that it is not enough for a learner to perform well against a set of non-strategic opponents (e.g., Shoham and Leyton-Brown 2008). It is also not enough that a learner converges to Nash equilibrium of the game which is being learned — which we call the “base game” — in self-play (e.g., Bowling and Veloso 2001, Conitzer and Sandholm 2007, Balduzzi et al. 2018, Letcher et al. 2018), as this does not guarantee that a player cannot benefit by deviating from that profile (i.e., submitting a different learning algorithm).

2. Repeated and stochastic games have many Nash equilibria. This one of the lessons of the folk theorems (e.g., Fudenberg and Maskin 1986), which say that any payoff profile in which each agent gets at least what they would get if they were being maximally punished by their counterparts is attained in some equilibrium. Narrowing the space of learning equilibria to those in which a reasonable welfare function is optimized improves the situation. (While we do not pursue the possibility here, there is some hope that the equilibrium selection problem in this setting could be addressed by pairs of learning algorithms which learn to agree on a welfare function even if they start out optimizing different welfare functions; cf. Van Damme 1986’s study of higher-order bargaining over welfare functions.) Moreover, our construction draws attention to the advantages of coordination by the principals on the learning algorithms used by the agents they deploy. The equilibrium selection problem can be solved if a welfare function is agreed upon beforehand, even if the environment in which the reinforcement learners will be deployed is initially unknown. In this sense, our framework improves upon results like that of Letcher et al. 2018, who guarantee that their opponent-aware algorithm converges to something like a local Nash equilibrium in self-play, but provide no guarantees on the quality (with respect to Pareto optimality or social welfare) of that solution. If principals are already in a setting where they can coordinate on the learning algorithms they deploy, they can do better by taking the LE approach and ensuring that these learning algorithms converge to a reasonable equilibrium.

Learning equilibrium was first discussed (under that name) by Brafman and Tennenholtz 2003. Their setting differs in several ways from ours, however. For stochastic games, their approach involves first finding a cooperative policy profile offline, and then deploying learning algorithms which enforce the usage of the cooperative policies. As we are interested in the setting where the environment is initially unknown (and therefore an initial offline planning step is not available), we instead consider online reinforcement learning in which learning consists in incremental stochastic updates towards the optimizer of a value function. Second, unlike Brafman and Tennenholtz we are motivated by a normative concern with constructing learning equilibria that are optimal with respect to an appropriate welfare function.

Several authors have recently studied cooperation among deep reinforcement learners. Peysakhovich and Lerer 2017, Lerer and Peysakhovich 2017, and Wang et al. 2018 all study the setting in which agents are able to plan offline, and train punishment policies to deter defection. For these authors, cooperation corresponds to policies which together maximize the sum of the players’ payoffs; the language of this paper, this means that they are enforcing the optimization of a particular (utilitarian) welfare function. In the online learning setting, Zhang and Lesser 2010, Foerster et al. 2018, and Letcher et al. 2018 have studied “opponent-aware learning”, in which players can see the parameters of their counterparts’ policies. The learning algorithms of Foerster et al. and Letcher et al. are “opponent-shaping” in that they choose parameter updates which are intended to influence their counterparts’ parameter updates in a way that improves their own payoffs. Foerster et al. find that their learning algorithm, learning with opponent-learning awareness (LOLA), leads to higher rates of cooperation in iterated Prisoner’s Dilemmas than naive learners. And, Letcher et al. show that a certain combination of LOLA and Zhang and Lesser 2010’s “lookahead” learning algorithm guarantees convergence to a stable fixed point, when such a point exists. However, as discussed in points 1 and 2 above, this is insufficient to guarantee that either the learning algorithms themselves constitute a learning equilibrium, or that the stable fixed point which is converged to constitutes a reasonable pair of policies (in the sense of scoring high on some measure of total welfare).

1See also the survey of cooperation in deep reinforcement learning in Hernandez-Leal et al. 2019 Section 3.5].
We work with stochastic games [Littman, 1994], a generalization of Markov decision processes to the multi-agent setting. We will focus on the case in which the principals can coordinate on a random seed (or make their policy parameters visible, as in the literature on opponent-awareness). This means that an agent always knows with certainty whether its counterpart’s actions are consistent with a cooperative learning algorithm. We will refer to this as “perfect policy monitoring”. We also present initial experiments in which this doesn’t hold (and therefore players must infer whether their counterpart is following the cooperative optimization schedule). We show how to construct learning equilibria by punishing deviations from the optimization of the welfare function in the perfect policy monitoring case. These equilibria involve finite-time punishments. The duration of punishment phases depends on the amount of data it takes to accurately estimate a punishment policy and on the amount of time it takes for the finite-horizon value of the learned punishment policy to approach its infinite-horizon average value. Thus, the theory presented here differs from other theoretical work in similar settings (e.g. Lerer and Peysakhovich 2017, Letcher et al. 2018) by making explicit how the enforcement of cooperation depends on the mixing time of the stochastic game and the accuracy in the estimation of the punishment policy.

2 Learning equilibrium in stochastic games

Consider a situation in which multiple agents interact with their environment over time, incrementally updating the policies which they use to choose actions. These updates are controlled by their learning algorithms, which are fixed by their principals before the agents begin interacting with the environment. It is helpful to distinguish between three different games that are being played here:

- The **base game**, which is defined by the dynamics of the environment, the reward functions of the agents, and the policies available to each agent. In this paper, the “base game” will be a stochastic game (described below). We can ask whether a particular profile of policies is a Nash equilibrium in the base game. However, as discussed in point 1 in the introduction, this would not guarantee that the learning algorithms which converge to that profile of policies are in equilibrium.

- The **learning game**, in which the principals simultaneously submit learning algorithms to be deployed in the base game environment, and attain payoffs equal to the limit of the average rewards they generate.

- The **update games** at each time \( t \), in which each player simultaneously decides which policy they will follow at time \( t+1 \) based on their history of observations. In our construction of learning equilibria, We will define cooperation in the update game as (roughly) taking a stochastic policy update step towards the optimum of the welfare function. Acting otherwise will be called defection.

We work with stochastic games [Littman, 1994], a generalization of Markov decision processes to the multi-agent setting. We will assume only two players, \( i, j \). In a stochastic game, players simultaneously take actions at each time \( t \in \mathbb{N} \); observe a reward depending on the current state; and the state evolves according to a Markovian transition function. Formally, a 2-player stochastic game is a tuple \((S, A_1, A_2, r_1, r_2, P)\), where

- \( S \) is the state space, with the state at time \( t \) denoted \( S_t \);
- \( A_i \) is the space of actions (assumed to be finite) available to player \( i \), with the action taken by player \( i \) at time \( t \) denoted as \( A_i^t \);
- \( r_i \) is the reward function of player \( i \), with the reward gained by player \( i \) at time \( t \) given by \( R_i^t = r_i(S, A_1^t, A_2^t) \);
- \( P \) is a Markovian transition function, such that the conditional distribution over states at time \( t + 1 \) is given by \( P(S_{t+1} \in B \mid (S^t, A_1^t, A_2^t)_{t=0}^t) = P(S_{t+1} \in B \mid S^t, A_1^t, A_2^t) \) for measurable \( B \subseteq S \).

Define a (stationary) policy \( \pi_i \) to be a mapping from states to random variables taking values in \( A_i \), such that \( P\{\pi_i(s) = a\} \) is the probability that player \( i \) takes action \( a \) at state \( s \) when following policy \( \pi_i \). Players wish to learn a policy which (in some sense) leads to large cumulative reward. In the online setting, each player is learning over a space of policies \( \Pi_\Theta = \{\pi_\theta : \theta \in \Theta\} \) for some
parameter space \( \Theta \). Correspondingly, define for parameter profiles \( \theta = (\theta_1, \theta_2) \) (and arbitrary initial state \( s^0 \)) the policy-value functions

\[
V_i(\theta) = \lim_{T \to \infty} T^{-1} \mathbb{E}_{\pi_{\theta_1}, \pi_{\theta_2}} \left( \sum_{t=1}^{T} R_t^i \mid S^0 = s^0 \right),
\]

where \( \mathbb{E}_{\pi_{\theta_1}, \pi_{\theta_2}} \) denotes the expectation with respect to trajectories in which players follow policies \( \pi_{\theta_1}, \pi_{\theta_2} \) at each time step. That is, \( V_i(\theta) \) is player \( i \)'s expected average return when player \( i \) follows \( \pi_{\theta_i} \) and player \( j \) follows \( \pi_{\theta_j} \). This limit is guaranteed to exist by the ergodicity assumption which is made in our construction of learning equilibria (Theorem 3.1 below).

Let \( H^t \) be the history of observations until time \( t \). We will assume that both agents fully observe the state and each others’ actions and rewards, such that \( \{(S^0, A^1_t, A^2_t, R^1_t, R^2_t)\}_{t=0}^T \in H^T \).

Define the value to player \( i \) of learning algorithm profile \( \sigma = (\sigma_1, \sigma_2) \) starting from history \( H^t \) as

\[
V_i(H^t, \sigma) \lim \inf_{T \to \infty} T^{-1} \mathbb{E}_\sigma \left( \sum_{t=0}^{T} R^i_t \mid H^t \right),
\]

where \( \mathbb{E}_\sigma \) is the expectation taken with respect to trajectories in which agents follow learning algorithms \( \sigma_1, \sigma_2 \) at each step. Let the initial history be \( H^0 = \{(S^0, A^1_0, A^2_0)\} \). Then, the game described above may be written in normal form as a game with strategy spaces \( \Sigma_1, \Sigma_2 \) corresponding to spaces of possible learning algorithms, and player \( i \)'s utilities at each profile \( (\sigma_1, \sigma_2) \) given by \( V_i(H^t, \sigma_1, \sigma_2) \). This a learning game. A learning equilibrium is a Nash equilibrium of a learning game:

**Definition 2.1** (Learning equilibrium). Learning algorithm profile \( (\sigma_1, \sigma_2) \) is an learning equilibrium of the learning game with learning algorithm spaces \( \Sigma_1, \Sigma_2 \) if

\[
\sup_{\sigma_1' \in \Sigma_1} V_1(H^0, \sigma_1', \sigma_2) \leq V_1(H^0, \sigma_1, \sigma_2) \quad \text{and} \quad \sup_{\sigma_2' \in \Sigma_2} V_2(H^0, \sigma_1, \sigma_2') \leq V_2(H^0, \sigma_1, \sigma).
\]

Lastly, we classify histories based on whether agents can verify whether their counterpart’s actions are consistent with a particular learning algorithm.

**Definition 2.2** (Learning algorithm). A learning algorithm for player \( i \) is a mapping \( \sigma_i \) from histories \( H^t \) to parameters, i.e., \( \sigma_i(H^t) \in \Theta \).

**Definition 2.3** (Policy monitoring). Let \( \chi^i_t = 1 \left[ A^i_t = \pi_{\sigma_i}(H^t) \right] \). The set of histories \( \mathcal{H} \) satisfies perfect policy monitoring for algorithms \( \sigma_1, \sigma_2 \) if for each \( H = \{H^0, H^1, \ldots\} \in \mathcal{H} \) each \( t \),

\[
\{\chi^1_t, \chi^2_t\}_{t=1}^T \in H^T. \tag{1}
\]

\( \mathcal{H} \) exhibits imperfect policy monitoring if (1) is not satisfied for some \( H^t \).

Opponent-aware algorithms [Zhang and Lesser 2010, Foerster et al. 2018, Letcher et al. 2018] are an example of learning algorithms which make use of perfect policy monitoring.

### 3 Enforcement of welfare-optimal learning

The learning algorithms we construct will converge to the optimizer of a welfare function in self-play. The welfare function is intended to encode a compromise between the principals’ individual payoffs. One intuitive welfare function is the utilitarian welfare (sometimes just called the “social welfare”). For player utility functions \( u_i \) and outcomes \( x \), this function calculates welfare according to \( u^{\text{util}}(x) = u_1(x) + u_2(x) \). Another welfare function is the Nash welfare [Nash 1950]. This welfare function also depends on “disagreement” utilities \( u^d_i \), which represents the value a player might get if the parties fail to compromise. (For instance, [Felsenthal and Diskin 1982] argue that the disagreement point for a player should be the minimum utility they receive among Pareto-optimal outcomes.) Nash showed that the only way of deciding on an outcome \( x \) which satisfied some natural axioms was

\[
\arg \max_{x} u^{\text{Nash}} \{u_1(x), u_2(x)\}, \text{ where } u^{\text{Nash}} \{u_1(x), u_2(x)\} = \{u_1(x) - u^d_1\} \times \{u_2(x) - u^d_2\}.
\]
Let \( w \) be the social welfare function. Let \( \theta \) be the parameters of the welfare function, which means that all players are weakly better off when there are more resources to go around. The table below contains a discussion of widely-discussed welfare functions from the social choice and cooperative bargaining literatures.

<table>
<thead>
<tr>
<th>Name of welfare function ( w )</th>
<th>Form of ( w { V_1(\theta), V_2(\theta) } )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nash [Nash, 1950]</td>
<td>( [V_1(\theta) - V_1^d] \cdot [V_2(\theta) - V_2^d] )</td>
</tr>
<tr>
<td>Kalai-Smorodinsky [Kalai and Smorodinsky, 1975]</td>
<td>( -\ell \left{ \frac{V_2(\theta) - V_2^d}{V_2(\theta) - V_2^d} = \sup_{\theta_1, \theta_2} V_1(\theta_1, \theta_2) - V_2^d \right} \sup_{\theta_1, \theta_2} V_2(\theta_1, \theta_2) - V_2^d )</td>
</tr>
<tr>
<td>Egalitarian [Kalai, 1977]</td>
<td>( \min \left{ V_1(\theta) - V_1^d, V_2(\theta) - V_2^d \right} )</td>
</tr>
<tr>
<td>Utilitarian (e.g. Harsanyi, 1955)</td>
<td>( V_1(\theta) + V_2(\theta) )</td>
</tr>
</tbody>
</table>

Table 1: Welfare functions, adapted to the multi-agent RL setting where two agents with value functions \( V_1, V_2 \) are bargaining over the pair of policies with parameters \( \theta = (\theta_1, \theta_2) \) to deploy. The function \( \ell \) in the definition of the Kalai-Smorodinsky welfare is the \( \infty \)-0 indicator, used to enforce the constraint in its argument.

Note that in bargaining problems in which the set of feasible payoffs is convex, the Nash welfare function uniquely satisfies the properties of (1) Pareto optimality, (2) symmetry, (3) invariance to affine transformations, and (4) independence of irrelevant alternatives. See [Zhou, 1997] for a characterization of the Nash welfare in non-convex bargaining problems. The Nash welfare can also be obtained as the subgame perfect equilibrium of an alternating-offers game as the “patience” of the players goes to infinity [Binmore et al., 1986].

On the other hand, Kalai-Smorodinsky uniquely satisfies (1)-(3) plus (5) resource monotonicity, which means that all players are weakly better off when there are more resources to go around. The egalitarian solution instead satisfies (1), (2), (4), and (5). The utilitarian welfare function is implicitly used in [Brafman and Tennenholtz, 2003]'s initial work on learning equilibrium and the work of [Peynakovich and Lerer, 2017], [Lerer and Peynakovich, 2017], [Wang et al., 2018] on cooperation in sequential social dilemmas.

Table 1 contains a discussion of widely-discussed welfare functions from the social choice and cooperative bargaining literatures.

The welfare function \( w \) will act on policy profiles \( \theta \), i.e., write \( w(\theta) = w \{ V_1(\theta), V_2(\theta) \} \). Write the set of (locally) welfare-optimal policy profiles as \( \Theta^{2,C} \). As discussed below, we expect to be able to construct algorithms which converge to a policy profile in \( \Theta^{2,C} \) using standard reinforcement learning methods. However, in order to construct a learning equilibrium, we also need to be able to disincentivize defections from an algorithm which optimizes \( \theta \). For this reason, we introduce a punishment point \( V_j^p \) for player \( j \). To construct an equilibrium, we will need to assume that a player’s punishment payoffs are worse than their bargaining payoffs. That is, for each \( i \) there exists a punishment policy parameter \( \theta_i^p \) such that

\[
\max_{\theta_j} V_j(\theta_i^p, \theta_j) = V_j^p < \inf_{\theta \in \Theta^{2,C}} V_j(\theta). \tag{2}
\]

3.1 Construction of welfare-optimal learning equilibria

In this section, we show how to construct welfare-optimal learning equilibria by optimizing a welfare function and punishing deviations from the optimization of this welfare function. We focus on the case of perfect policy monitoring (Definition 2.3).

Let \( \{ \delta^t \}_{t=1}^\infty \) be a sequence of step sizes for the policy parameter updates. Let \( \hat{w} \) be an estimator of the welfare function. Let \( \mathcal{O}^C \) be a base policy learning algorithm which maps histories \( H^t \) to candidate updates \( (\theta^t_1, \theta^t_2) \). For instance, \( \mathcal{O}^C \) might be a vanilla policy gradient algorithm

\[
\mathcal{O}^C(H^t) = \theta^t + \delta^t \nabla \hat{w}(\theta^t).
\]

Define \( \theta^C = \arg \max_{\theta} w(\theta) \) and \( \theta_i^p = \arg \max_{\theta_j} V_j(\theta_i^p, \theta_j) \). Write the updated parameters returned by \( \mathcal{O}^C \) as \( \theta_i^{C,t+1} = \mathcal{O}^C(H^t) \). We say that player \( i \) defects at time \( t \) if \( \theta_i^t \neq \theta_i^{C,t} \), and construct a strategy which punishes defections. The learning equilibrium construction presented here divides time points into blocks, and deploys either a cooperative or punishment learning algorithm in each block depending on whether there was a defection in the previous block. Note that, while equilibrium
could be accomplished with a "grim-trigger" strategy which never stops punishing after a defection, we deliberately use limited-time punishments to make our construction more forgiving. (This block construction is similar to that used in [Wiseman, 2012]'s construction of equilibria for repeated games in which players only see noisy observations of the underlying stage game.) Specifically, in the $b^{th}$ block of length $M^b + N^b$:

1. If the other player cooperated at each time point in the previous block, then

   - Deploy cooperative policy learner $O^C$ for $M^b$ steps to obtain a good estimate of the welfare-optimal policy $\theta_{b,C}^i$;
   - Deploy policy $\theta_{b,C}^i$ for $N^b$ time steps to generate payoffs close to $V_i^C$.

2. If the other player defected at any time point in the previous block, then

   - Deploy punishment policy learner $O^p$ for $M^b$ steps to obtain a good estimate of the punishment policy $\theta_{b,p}^i$;
   - Deploy policy $\theta_{b,p}^i$ for $N^b$ time steps to generate payoffs close to $V_i^p$.

This algorithm is summarized in algorithmic form in Appendix A.

**Theorem 3.1** (Construction of forgiving and welfare-optimal learning equilibrium). Consider a learning game with perfect policy monitoring. Fix a welfare function $w$, cooperative policy optimizer $O^C$, and punishment policy optimizer $O^p$. Consider the corresponding learning algorithms $(\sigma_{1,LE}^i, \sigma_{2,LE}^j)$ constructed as in Algorithm 3.

Let $t^b = \sum_{b < b} (M^b + N^b)$, i.e., $t^b$ is the time at which the $b^{th}$ block starts. Write $V^{t,t'}(H^t, \theta) = (t' - t)^{-1} \mathbb{E}_\theta \left( \sum_{t=i}^{t'} R^v | H^t \right)$. For $i = 1, 2$, let $\theta^C_i \in \Theta^{2-C}$ index a (locally) welfare-optimal policy and let $\theta^p_i$ be a punishment policy parameter satisfying inequality $2$. Use a similar notation for the average value attained in over time points $t, \ldots, t'$ under combinations of stationary policies $\theta_i$ and learning algorithms $\bar{\sigma}_i$. For ease of notation, define the following quantities:

$$
\hat{V}_{j}^{b,C} = V^{t^b + M^b + t^b + M^b + N^b}(H^{t^b}, \theta_{i}^{b,C}, \theta_{j}^{b,C});
$$

$$
V_{j}^{b,C} = V^{t^b + M^b + t^b + M^b + N^b}(H^{t^b}, \theta_{i}^{C}, \theta_{j}^{C});
$$

$$
\hat{V}_{j}^{b,p} = V^{t^b + M^b + t^b + M^b + N^b}(H^{t^b}, \theta_{i}^{b,p}, \bar{\sigma}_{j});
$$

$$
V_{j}^{b,p} = V^{t^b + M^b + t^b + M^b + N^b}(H^{t^b}, \theta_{i}^{p}, \theta_{j}^{p}).
$$

Make the following assumptions, for $j = 1, 2$:

1. For some $\sigma_{C}^F \in (0, 1)$, $\hat{V}_{j}^{b,C}$ converges in probability to $V_{j}^{b,C}$ at a $(t^b)^{\sigma_{C}^F}$ rate;

2. For some $\sigma_{p}^F \in (0, 1)$, $\hat{V}_{j}^{b,p}$ converges in probability to $V_{j}^{b,p}$ at a $(t^b)^{\sigma_{p}^F}$ rate;

3. For a Markov chain indexed by policy profile with parameter $\theta$, let $P^t_{\theta}$ denote $t$ applications of its transition operator, and $\nu_0$ its stationary distribution. Then the stochastic game is uniformly ergodic (e.g. [Ormer, 2013]) in the sense that, for any measure over states $\nu$,

$$
\sup_{\theta} \| P^t_{\theta} - \nu_0 \| \leq K \gamma^t,
$$

for constants $K$ and $\gamma$;

4. For any measure $\nu$ and policy profile parameter $\theta$, $\nu$ and $\nu_0$ are dominated by measure $\mu$ and the reward function satisfies $\int |r_j(S, \pi_{\theta}(S)) | d\mu(S) \leq |R_j|_\mu < \infty$;

5. The reward function is bounded, such that the most by which player $j$'s average payoff in a given interval can exceed $V_{j}^{C}$ is bounded by $\Delta_j < \infty$;

6. $\sum_{b \leq B} M^b \left\{ \sum_{b \leq B} (M^b + N^b) \right\}^{-1} \rightarrow 0$.

Under Assumptions 1-6, $(\sigma_{1,LE}^i, \sigma_{2,LE}^j)$ is a learning equilibrium and $w(H^0, \sigma_{1,LE}^i, \sigma_{2,LE}^j)$ converges to a locally welfare-optimal policy profile.
Proof. In Appendix A.

It is beyond the scope of this article to analyze the conditions under which the assumptions of Theorem 3.1 hold. But, we note that Assumption 1 is a standard desideratum for learning algorithms in single-agent reinforcement learning. Assumption 2 is a standard desideratum in the case of zero-sum games, which can be used to construct a punishment policy by setting the punishment policy equal to the one which minimizes the other player’s best-case payoff. For instance, Yang et al. [2019] find that deep Q-network (DQN; Mnih et al. 2013) estimators converge to the optimal Q-function at a rate, where the optimal Q-function can be written as a composition of sparse functions which are Hölder smooth and depend on of their inputs. This result extends to a DQN variant of minmax Q-learning [Littman 1994], which can be used to construct a punishment policy.

Also note that several of the welfare functions in Table 1 straightforwardly inherit useful properties with respect to single-agent policy-learning methods. For instance, the utilitarian welfare function admits the standard Bellman equations, and because it is simply the sum of individuals’ value functions, has an unbiased gradient estimator whenever the individual value functions have an unbiased gradient estimator. As a sum of a monotone transformation of the value functions, the Nash welfare inherits improvement guarantees — for instance, those which form the theoretical background to trust region policy optimization [Schulman et al., 2015]. Also see Roijers et al. [2013] for a survey of multi-objective reinforcement learning, since the optimization of a welfare function can be seen as what these authors call a scalarization of the expected return (SER) approach to a multi-objective reinforcement learning problem.

Punishment policies can be constructed by minimizing the defector’s best-case payoffs, (temporarily) converting the base game to a zero-sum game. Compared with general-sum games, reinforcement learning for zero-sum games is relatively well-understood. Several algorithms with theoretical convergence guarantees exist for policy-learning in extensive form zero-sum games [Srinivasan et al., 2018; Lockhart et al., 2019], and model-free deep reinforcement learning has seen high-profile successes in complex zero-sum games [Vinyals et al., 2019; Schrittwieser et al., 2019].

Lastly, uniform ergodicity (Assumption 3) is a standard assumption in the theoretical analysis of reinforcement learning; see e.g. Ortner [2018] and references therein. Of course, this may be a highly unrealistic assumption. See Appendix B for discussion of an “ex interim” variant of the learning equilibrium solution concept, which drops the ergodicity assumption, making it applicable to a much wider class of environments (but losing guarantees on the payoffs generated by the learning algorithms).

3.2 Experiments

We now describe simulations illustrating the enforcement of steps towards the welfare-optimal policy, using the iterated Prisoner’s Dilemma (IPD) and the Coin Game environment [Lerer and Peysakhovich, 2017; Foerster et al., 2018]. We use a tit-for-tat-(TFT)-like class of learning algorithms. These learning algorithms maintain a cooperative network trained on the welfare function, a punishment network trained on the negative of the counterpart’s payoffs, and switch from cooperation to punishment when they detect a defection from the optimization of the welfare function.

We present experiments for the perfect policy monitoring (PM) setting. We also present some initial results from the imperfect policy monitoring (NoPM) setting, where the agents must infer whether their counterpart is cooperating.

Perfect policy monitoring: Iterated Prisoner’s Dilemma

Our variant of the IPD involves repeated play of the matrix game with expected payoffs as in Table 2. We treat this iterated game as a stochastic game in which the state at time \(t\) is given by the actions taken at the last time step, i.e., \(S^t = (A_1^{t-1}, A_2^{t-1})\). Player i’s policy \(\pi_\theta_i\) gives the probability of each action given player j’s action at the previous timestep. In particular, \(\theta_i = (\theta_{i,11}, \theta_{i,12}, \theta_{i,21}, \theta_{i,22})\), and

\[
P(\pi_\theta_i(S^t) = a) = P(\pi_\theta_i(A_1^{t-1}) = a) \propto \exp(\theta_{i,A_2^{t-1}}).
\]


We write the (expected) reward function as $r_i(a_i, a_j)$, dropping state argument because rewards are independent of state here. We introduce randomness by drawing rewards $R_i^t \sim N \{ r_i(A_i^t, A_j^t), 0.1 \}$.

For the base cooperative policy optimization algorithm $\mathcal{O}^C$ we use REINFORCE \cite{Williams:1992}. We use the utilitarian welfare function, $\omega(t)$. Thus, for episode length $u$,

\[
\theta_{i,t}^{C,t+u} = \mathcal{O}^C(H^{t+u}) = \theta_{i,t}^{C,t} + \delta^t \nabla_{\theta} \log [P\{\pi_{\theta_i}(S^t) = A_i^t\} P\{\pi_{\theta_j}(S^t) = A_j^t\}] \sum_{v=t}^{t+u} (R_i^v + R_j^v)
\]

Similarly, we update $\theta_{i,t}^{P,t}$ via REINFORCE steps on rewards $-R_j^v$ collected during punishment episodes. Finally, player $i$ maintains a policy network with parameter $\theta_{i,t}^{C,t}$ which is updated according the cooperative update rule for player $j$. This is done to compare player $j$'s actual policy to what their policy would be if they were cooperating. Now, policy visibility is actually not necessary for implementing strategies under perfect policy monitoring (only agreement on a random seed is). But for simplicity we activate punishment when $||\theta_j^t - \theta_{i,j}^{C,t}|| > 0.01$. When player $i$ detects a defection, they follow the punishment policy for 5 episodes and then return to the cooperative policy. We refer to this learning algorithm as $\sigma_i^{TFT-PM}$.

Figure 2 summarizes the performance of $\sigma_i^{TFT-PM}$ played against itself in the IPD described above. As expected, $\sigma_i^{TFT-PM}$ converges to mutual cooperation. When played against a naive REINFORCE learner, $\sigma_i^{TFT-PM}$ correctly punishes, resulting in mutual defection.

**Perfect policy monitoring: Coin Game**

In the Coin Game environment, depicted in Figure 3, a Red and Blue player navigate a grid and pick up randomly-generated coins. Each player gets a reward of 1 for picking up a coin of any color. But, a player gets a reward of -2 if the other player picks up their coin. The state space consists of encodings of each player’s location and the location of the coin.

As before, $\sigma_i^{TFT-PM}$ maintains a cooperative network trained on reward function $R_i^v + R_j^v$ and a punishment network trained with rewards $-R_j^v$. These networks are trained with proximal policy optimization (PPO) \cite{Schulman:2017}. And as before $\sigma_i^{TFT-PM}$ switches from the cooperative to the punishment policy when $||\theta_j^t - \theta_{i,j}^{C,t}|| > 0.01$.

$\sigma_i^{TFT-PM}$ again converges to mutual cooperation in self-play (picking own coin 100% of the time). This is significantly better than the performance of naive PPO in self-play (Coin Game baseline), which does not converge to mutual cooperation. And against a naive PPO learner, $\sigma_i^{TFT-PM}$ successfully punishes defection, though at the expense of very low payoffs for itself.

**Imperfect policy monitoring: Iterated Prisoner’s Dilemma**

We also ran some exploratory experiments for the imperfect policy monitoring setting. At each step, each player compares the likelihood of their counterpart’s actions under the cooperative sequence of parameter updates to the maximum likelihood stationary policy. This stationary policy estimate represents the hypothesis that the counterpart is not following the cooperative learning algorithm. If the likelihood of the maximum likelihood stationary policy is sufficiently higher than that of the

---

These experiments were run using a different punishment policy and a different, off-policy version of REINFORCE than was used in the PM experiments reported above. This will be changed in the final version of the paper to make the methodology consistent.
cooperative sequence of updates, the estimated minimax action is played. We construct a test by bootstrapping [Efron 1992] the log-likelihoods of the observed actions under the cooperative and maximum-likelihood stationary parameter sequences. See Appendix C for details.

Refer to this learning algorithm as $\sigma^{TFT-NoPM}_i(H^t)$. The actions of a player following $\sigma^{TFT-NoPM}_i(H^t)$ are given by:

$$A^t_i \sim \begin{cases} 
\pi_{\theta,C}(S^t), & \text{if hypothesis that counterpart is cooperating is not rejected,} \\
\min_{a_i} \max_{a_j} \hat{r}_j(a_i, a_j), & \text{otherwise.}
\end{cases}$$

Figure 3 shows the results of $\sigma^{TFT-NoPM}$ in self-play. This learning algorithm profile is still able to learn the mutual-cooperation policy profile. On the other hand, our hypothesis test correctly detects defections by a naive learner, resulting in mutual defection in the base game.

We expect that Theorem 3.1 can be extended straightforwardly to the imperfect policy monitoring case by using such a procedure for deciding when to punish, assuming that the errors made by the hypothesis test used to detect defection decay sufficiently quickly as data accumulates.

4 Discussion

We have sketched a framework for understanding rationality and cooperation on part of the principals of reinforcement learners. It is of paramount importance that powerful machine intelligences avoid conflict when they are eventually deployed. One way to avoid such outcomes is to construct profiles of strategies which rational actors prefer to profiles that lead to conflict. We have argued that the appropriate notion of rationality in this context is learning equilibrium (rather than mere convergence to Nash equilibrium, for instance). We have taken initial steps towards showing how learning equilibria which are cooperative — in the sense of optimizing a reasonable welfare function — can be constructed.

Many open questions remain. First, while we have relied on a welfare function, we have said little about which welfare function should be used. As alluded to in point 2 in the introduction, the ideal scenario would be for the principals of the reinforcement learning systems in question to coordinate on a welfare function before deploying these systems. Another direction worth exploring is the design of learning algorithms which detect whether their counterparts are optimizing a welfare function other than their own, and adjust accordingly to optimize some compromise welfare function.

Our framework is limited in many ways. We are currently working to construct learning equilibria in the imperfect policy monitoring case using testing procedures like that described in Section 3.2 and Appendix C. Beyond that, the theory should be extended to the case of partial state observability, more than two agents, and criteria other than average-reward. We also have assumed that the agents’ reward functions are known, but a complete framework should address the problem of incentives to misrepresent one’s reward function to improve one’s payoffs in the welfare-optimal policy profile. This raises questions of mechanism design; see [Nisan et al. 2007, Ch. 9] for an introduction aimed...
Figure 2: Performance of different learning algorithms in iterated Prisoner’s Dilemma (IPD) and Coin Game. For IPD prob axis is the probability of mutual cooperation (cc) and defection (dd) under the learned policy profile at the corresponding time step, and the payoffs axis is the reward for each player at the corresponding time steps. For Coin Game, the top plots display the probability of each agent picking their own coin (i.e. cooperating), and the fraction of steps into the game the agent picks a coin. Coin Game baseline refers to naive PPO self-play.

at computer scientists. Note that the use of mechanisms in which principals truthfully reveal their reward functions may greatly restrict the set of available welfare functions. It is well-known, for instance, that the Vickrey-Clarke-Groves family of mechanisms — which implement the utilitarian welfare function — is the only family of mechanisms which are both efficient and truthful [Green and Laffont, 1977, Holmström, 1979].

In terms of implementation, it may be necessary to develop novel reinforcement learning methods tailored for the optimization of different welfare functions. For instance, we have focused on policy-based reinforcement learning, but the development of value-based methods would likely broaden the space of practical algorithms available for implementing our approach. This is nontrivial, as the welfare functions in Table 1 other than the utilitarian welfare do not admit a Bellman equation (see
Figure 3: Performance of $\sigma_{\text{TFT}-\text{NoPM}}$ against itself (left) and against a naive REINFORCE learner (right) in the iterated Prisoner’s Dilemma. The prob axis is the probability of mutual cooperation (cc) and mutual defection (dd) under the learned policies $\theta^t$ at the corresponding time step, and the payoffs axis is the reward for each player at the corresponding time steps. These values are averaged over 10 replicates.

the discussion of nonlinear scalarization functions in [Roijers et al., 2013]’s review of multi-objective reinforcement learning. The question of computational feasibility may also interact in fruitful ways with the problem of selecting a welfare function. Computational considerations may eliminate welfare functions, such as Kalai-Smorodinsky, which are much harder to optimize than others.

Among the most serious limitations of our framework are the ergodicity assumptions used to construct equilibria. Ergodicity is not a realistic assumption about the real world. A more realistic framework might involve “ex-interim” learning equilibrium concepts, such as that sketched in Appendix B.

Acknowledgements

Many thanks to Tobias Baumann, Flo Dorner, Jakob Foerster, Daniel Kokotajlo, and Johannes Treutlein for helpful comments on drafts of this document.

References


Appendix A: Proof of Theorem 3.1

Proof. For brevity, write $V_j^C(\theta^C) = V_j^{C,b}$ and $V_j^P(\theta^P) = V_j^{p,b}$. Let $\chi^b$ indicate whether player $j$ cooperated in block $b$ (setting $\chi^{-1} = 1$).

Notice that, under Assumption 5, we can bound the difference between $V_j^{C,b}$ and the finite-time average reward under the cooperative policy in block $b$ as follows:

$$V_j^{b,\theta^b+M^b+N^b}(H^{b},\sigma_i^C,\sigma_j^C) - V_j^{C,b} \leq (M^b + N^b)^{-1} \left\{ M^b \Delta_j + N^b \left( \bar{V}_{j,C}^b - V_j^{C,b} \right) \right\}.$$
And by Assumptions 1, 3, and 4, there exist constants $K^C$, $r_{b}^{C}$, and $r_{c}^{C}$ such that

$$\hat{V}^{b,C} - V^{C} = \left(\hat{V}^{b,C} - V^{b,C}\right) + \left(V^{b,C} - V^{C}\right)$$

$$\leq (t^{b})-r_{c}^{C} P \left\{ \hat{V}^{b,C} - V^{b,C} < (t^{b})-r_{c}^{C} \right\}$$

$$+ \Delta_{2} P \left\{ \hat{V}^{b,C} - V^{b,C} > (t^{b})-r_{c}^{C} \right\}$$

$$+ \left(N^{b}\right)^{-1} \sum_{t=t^{b}+M^{b}}^{t^{b}+M^{b}+N^{b}} \int_{t^{b}} \{ \pi_{\theta^{b},C}(S^{t}) \} \left\{ dP^{\theta^{b},C}(S^{t}) - d\nu_{0}(S^{t}) \right\}$$

$$\leq (t^{b})-r_{c}^{C} \left\{ 1 - (t^{b})-r_{c}^{C}(r_{c}^{E}-r_{c}^{C}) K^{C} \right\}$$

$$+ \Delta_{2}(t^{b})-r_{c}^{E}(r_{c}^{E}-r_{c}^{C}) K^{C}$$

$$+ K|R_{j}|_{\mu} \left\{ N^{b}(1 - \gamma) \right\}^{-1} \left( 1 - \gamma^{-N^{b}} \right)$$

By Assumptions 2, 3, and 4 the same holds for the corresponding punishment point quantities, making the necessary changes. Moreover, by Assumption 6,

$$\left(t^{B}\right)^{-1} \sum_{b \leq B} \left( 1 - \gamma^{-N^{b}} \right) \leq \left(t^{B}\right)^{-1} B$$

$$\rightarrow 0.$$

We will now write the value attained by player 2 in the first $B$ blocks in a way that allows us to apply these convergence statements. Then we will take expectations and apply Assumption 3 to show that each term goes to 0. Letting $\Omega$ be a set of measure 1 and $\chi^{b}(\omega)$ the value of $\chi^{b}$ along an arbitrary sample path $\omega \in \Omega$, write $B^{D} = \sup_{\omega \in \Omega} \limsup_{B \rightarrow \infty} B^{-1} \sum_{b \leq B} \left\{ 1 - \chi^{b}(\omega) \right\}$. And, write

$$\mathbb{1}_{<c}^{b,C} = \mathbb{1} \left\{ \hat{V}^{b,C} < V^{b,C} + (t^{b})-r_{c}^{C} \right\};$$

$$\mathbb{1}_{<p}^{b} = \mathbb{1} \left\{ \hat{V}^{b,p} < V^{b,p} + (t^{b})-r_{p}^{C} \right\}.$$

14
Thus, \( V_j^{t^B} - V_j^C = (t^B)^{-1} \sum_{b=0}^{B-1} (t^{b+1} - t^b) \mathbb{E}(\tilde{V}_j^b - V_j^C) \)
\[ \leq (t^B)^{-1} \sum_{b=0}^{B-1} \left[ M^b \Delta_j + N^b \mathbb{E} \left\{ \chi_b^{-1} \left( \tilde{V}_j^{b,C} - V_j^C \right) + \left( 1 - \chi_b^{-1} \right) \left( \tilde{V}_j^{b,p} - V_j^C \right) \right\} \right] \]
\[ = (t^B)^{-1} \sum_{b=0}^{B-1} \left[ M^b \Delta_j + N^b \mathbb{E} \left\{ \chi_b^{-1} \left( \tilde{V}_j^{b,C} - V_j^C \right) + \left( 1 - \chi_b^{-1} \right) \left( \tilde{V}_j^{b,p} - V_j^C \right) \right\} \right] \]
\[ = (t^B)^{-1} \sum_{b=0}^{B-1} \left[ M^b \Delta_j + N^b \mathbb{E} \left\{ \chi_b^{-1} \left( \tilde{V}_j^{b,C} - V_j^C \right) + \left( 1 - \chi_b^{-1} \right) \left( \tilde{V}_j^{b,p} - V_j^C \right) \right\} \right] \]
\[ + \mathcal{B}^D (V_j^p - V_j^C) \]
\[ \leq (t^B)^{-1} \sum_{b=0}^{B-1} M^b \Delta_j \]
\[ + (t^B)^{-1} \sum_{b=0}^{B-1} N^b \mathbb{E} \left\{ \left\{ (t^b)^{-r^p} + \rho(N^b) \right\} \mathbb{I}^b_{\leq C} + \Delta_j \left( 1 - \mathbb{I}^b_{\leq C} \right) \right\} \]
\[ + \mathcal{B}^D (V_j^p - V_j^C) \]
\[ = (t^B)^{-1} \sum_{b=0}^{B-1} M^b \Delta_j \]
\[ + (t^B)^{-1} \sum_{b=0}^{B-1} N^b \left[ \left\{ (t^b)^{-r^p} + \rho(N^b) \right\} \mathbb{I}^b_{\leq C} + \Delta_j \left( 1 - \mathbb{I}^b_{\leq C} \right) \right] \]
\[ + \mathcal{B}^D (V_j^p - V_j^C) \]
\[ \leq (t^B)^{-1} \sum_{b=0}^{B-1} M^b \Delta_j \]
\[ + (t^B)^{-1} \sum_{b=0}^{B-1} N^b \left[ \left\{ (t^b)^{-r^p} + \rho(N^b) \right\} \mathbb{I}^b_{\leq C} + \Delta_j \left( 1 - \mathbb{I}^b_{\leq C} \right) \right] \]
\[ + \mathcal{B}^D (V_j^p - V_j^C) \]
\[ \to \mathcal{B}^D (V_j^p - V_j^C). \]

Now, the quantity on the right hand side of in display \[3\] is bounded above by
\[ (t^B)^{-1} \sum_{b=0}^{B-1} M^b \Delta_j \]
\[ + (t^B)^{-1} \sum_{b=0}^{B-1} N^b \left[ \left\{ (t^b)^{-r^p} + \rho(N^b) \right\} \mathbb{I}^b_{\leq C} + \Delta_j \left( 1 - \mathbb{I}^b_{\leq C} \right) \right] \]
\[ + \mathcal{B}^D (V_j^p - V_j^C) \]
\[ \to \mathcal{B}^D (V_j^p - V_j^C). \]

Thus, \( V_j(H^t, \sigma_1, \sigma_j) \leq 0 \), with strict inequality if \( \mathcal{B}^D > 0 \).

Moreover, Assumption 1 implies that \( V_j(H^0, \sigma_1, \sigma_2) = V_j^C \).
Algorithm 1: Cooperative learning algorithm for player 1, $\sigma_1^C$

**Input:** Welfare function estimator $H^t \mapsto \hat{w}$, cooperative policy learner $O^C$, history $H^t$, current block index $b$, sub-block sizes $\{M^{b'}, N^{b'}\}_{b' \leq b}$

$p, b \leftarrow O$

/* Estimate cooperative policy */

for $t = \sum_{b' < b}(M^{b'} + N^{b'})$ to $\sum_{b' < b}(M^{b'} + N^{b'}) + M^b$ do

$A_t^1 \leftarrow \pi_{\theta_t^1}(S^t)$;

Observe $A_t^2$;

if $A_t^2 = \pi_{\theta_t^2}^{C,0}(S^t)$ and $\chi^b = 1$ then

$\chi^b \leftarrow 1$;

else

$S^{t+1} \sim P(\cdot | S^t, A_t^1, A_t^2)$;

$H^t \leftarrow H^{t-1} \cup \{A_t^1, A_t^2, R_1^t, R_2^t, S^{t+1}\}$;

end if

end for

return $H^t$, $\chi^b$

Algorithm 2: Punishment learning algorithm for player 1, $\sigma_1^P$

**Input:** Welfare function estimator $H^t \mapsto \hat{w}$, punishment policy learner $O^P$, history $H^t$, current block index $b$, sub-block sizes $\{M^{b'}, N^{b'}\}_{b' \leq b}$

/* Estimate punishment policy */

for $t = \sum_{b' < b}(M^{b'} + N^{b'})$ to $\sum_{b' < b}(M^{b'} + N^{b'}) + M^b$ do

$A_t^1 \leftarrow \pi_{\theta_t^1}(S^t)$;

Observe $A_t^2$;

if $A_t^2 = \pi_{\theta_t^2}^{P,0}(S^t)$ and $\chi^b = 1$ then

$\chi^b \leftarrow 1$;

else

$S^{t+1} \sim P(\cdot | S^t, A_t^1, A_t^2)$;

$H^t \leftarrow H^{t-1} \cup \{A_t^1, A_t^2, R_1^t, R_2^t, S^{t+1}\}$;

end if

end for

return $H^t$
Algorithm 3: Learning equilibrium strategy

**Input:** Welfare function estimator $H^t \mapsto \hat{w}$, cooperative policy learner $O^C$, punishment policy learner $O^p$

**Initialize** $\theta^0$, state $S^0$. for $b = 0, 1, \ldots$ do

- if $\chi^{b-1} = 1$ then
  - $H^t, \chi^b \leftarrow \text{Algorithm1}(H^{b-1})$;
- else if $\chi^{b-1} = 0$ then
  - $H^t \leftarrow \text{Algorithm2}(H^{b-1})$;
  - $\chi^b = 1$;

Appendix B: *Ex interim* learning equilibrium

The ergodicity assumption We used to construct learning equilibria is not realistic for the kinds of environments generally intelligent agents might be deployed in. Here We propose a solution concept which is applicable in general environments. The solution concept applies to agents for whom Bayesian planning is intractable. However, we lose the *ex post* guarantees on the agents’ long-run payoffs.

In (model-based) *ex interim* learning equilibrium, the agents’ decisions at time $t$ are obtained by planning against an estimated model $\hat{M}_t$. The cooperative policy at each time step is the one that maximizes the estimate of the welfare function obtained using the estimated model $\hat{M}_t$. Then, the cooperative action is just the action recommended by this cooperative policy. If an agent defects at time $t$, they are punished according to the their minimax value under estimated model $\hat{M}_t$.

We do not specify conditions on the estimator $\hat{M}$. The idea is that, in practice, the principals must agree on what a reasonable belief-forming process looks like before deploying their agents. One natural requirement is that the estimator be consistent, i.e., that it converge to the true environment dynamics as data accumulates. However, as We have not specified the class of generative models, even consistency might be too strong a requirement. Nevertheless, the hope is that this framework will least allow us to say:

1. If the estimator is consistent, then an *ex interim* learning equilibrium is indeed an *ex post* learning equilibrium — that is, it is a learning equilibrium with respect to the principals’ true payoffs under the submitted learning algorithms;
2. Even if the model is misspecified, agents playing an *ex interim* learning equilibrium are at least playing efficiently with respect to their “best guess” about the environment.

Define $w(H^t, \theta; \mathcal{M})$ to be the welfare under policy profile indexed by $\theta$ starting at history $H^t$ when $\mathcal{M}$ is the true generative model. Similarly define value functions $V_i(H^t, \theta; \mathcal{M})$. Then here is the formal setup:

**Definition 4.1 (Ex interim learning equilibrium).** Let $\hat{\mathcal{M}}$ be a function from histories to dynamics models (for instance, models of the transition dynamics of a stochastic game). Let $\hat{\mathcal{M}}^t = \hat{\mathcal{M}}(H^t)$ be an estimator of the dynamics at time $t$. Define the cooperative policy profile and corresponding action profile at time $t$ as

$$\theta^{C,t} = \arg \max_{\theta} w(H^t, \theta; \hat{\mathcal{M}}^t)$$

$$A^{C,t}_i = \pi_{\theta^{C,t}}(H^t)$$

\footnote{While this might seem unsatisfying, the need for agreement on other parameters (the payoffs, the common prior, the transition dynamics of the stochastic game, equilibrium selection principle, etc.) has has always been present in game theory.}

\footnote{For simplicity we assume deterministic policies (or stochastic policies in cases where the random seed is shared between agents).}
Define the punishment policy at history $H^t$ as
\[
\theta_{i}^{p,t} = \min_{\hat{\theta}_i} \max_{\theta_j} V_j(H^t, \theta_i, \theta_j; \hat{M}^t)
\]

Player $i$’s punishment policy is activated at time $t$ if $A_{j}^t \neq A_{j}^{C,t}$. Call $(\sigma_1, \sigma_2)$ an \textit{ex interim} learning equilibrium if, for each history $H^t$ and $i = 1, 2$
\[
\sigma_i(H^t) = \begin{cases} 
A_{i}^{C,t}, & \text{if punishment policy not active;} \\
\pi_{\theta_{i}^{p,t}}(H^t), & \text{otherwise.}
\end{cases}
\]

What justifies calling $(\sigma_1, \sigma_2)$ as defined above an \textit{equilibrium}? At each time step, player $i$ does not expect to benefit by choosing an action other than that recommended by $\sigma_i$, according to the common model estimate $\hat{M}^t$ — thus the “\textit{ex interim}”. So, if the principals were willing to agree at the outset that $\hat{M}$ is a reasonable way for their agent to form beliefs about the world, then it seems natural to make playing such an equilibrium a requirement of individual rationality. Still, there are obvious complications: for instance, what happens when the agents also have private signals which they can use to update their beliefs?

Lastly, we do not actually want to advocate for the use of a minimax punishment strategy (display 4). Another important direction is working out more forgiving strategies which still adequately discourage exploitation.

\textbf{Appendix C: Test for cooperativeness under imperfect policy monitoring}

We construct a procedure by which player $i$ can test whether player $j$ is playing the cooperative learning algorithm. Let $\theta_{j}^{D,t} = \arg \max_{\theta_j} \prod_{v=1}^{t} P\{\pi_{\theta_j}(S^v) = A_{j}^v\}$ be the maximum likelihood parameter under the hypothesis that player $j$ is playing a stationary policy; this will represent the hypothesis that player $j$ is defecting. We will use a bootstrap estimator of the sampling distribution of these log likelihoods to construct our test. To specify a hypothesis test, define the log likelihoods under cooperative and defecting learning algorithms (conditional on the sequence of states in $H^t$):
\[
\ell_{j}^{C,t} = \sum_{v=1}^{t} \log P\{\pi_{\theta_{j}^{C,v}}(S^v) = A_{j}^v\};
\]
\[
\ell_{j}^{D,t} = \sum_{v=1}^{t} \log P\{\pi_{\theta_{j}^{D,v}}(S^v) = A_{j}^v\}.
\]

For $b = 1, \ldots, 50$, we then:
1. Draw a list $H^{b}$ of 20 $(S^v, A_{j}^v)$ pairs with replacement from the previous 20 observations;
2. Compute a bootstrapped maximum likelihood stationary policy parameter from $H^{b}$, which we will call $\theta_{j}^{D,b}$, and obtain the corresponding log likelihood $\ell_{j}^{D,b}$;
3. Compute the log probabilities of each element of $H^{b,t}$ under cooperative policy sequence $\pi_{\theta_{j}^{C,v}}$ and obtain the corresponding log likelihood $\ell_{j}^{C,b,t}$;
4. Take the difference in log likelihoods $\Delta_{j}^{b,t} = \ell_{j}^{C,b,t} - \ell_{j}^{D,b}$.

This results in a bootstrap distribution of values $\Delta_{j}^{b,t}$. If the 95th percentile of this distribution is less than 0, player $i$ rejects the hypothesis that player $j$ is cooperating, and punishes accordingly.