Emergent Cooperation from Mutual Acknowledgment Exchange

Thomy Phan
Felix Sommer
Philipp Altmann
Fabian Ritz
LMU Munich
Munich, Germany
thomy.phan@ifi.lmu.de

Lenz Belzner
Technische Hochschule Ingolstadt
Ingolstadt, Germany

Claudia Linnhoff-Popien
LMU Munich
Munich, Germany

ABSTRACT

Peer incentivization (PI) is a recent approach, where all agents learn to reward or to penalize each other in a distributed fashion which often leads to emergent cooperation. Current PI mechanisms implicitly assume a flawless communication channel in order to exchange rewards. These rewards are directly integrated into the learning process without any chance to respond with feedback. Furthermore, most PI approaches rely on global information which limits scalability and applicability to real-world scenarios, where only local information is accessible. In this paper, we propose *Mutual Acknowledgment Token Exchange (MATE)*, a PI approach defined by a two-phase communication protocol to mutually exchange acknowledgment tokens to shape individual rewards. Each agent evaluates the monotonic improvement of its individual situation in order to accept or reject acknowledgment requests from other agents. MATE is completely decentralized and only requires local communication and information. We evaluate MATE in three social dilemma domains. Our results show that MATE is able to achieve and maintain significantly higher levels of cooperation than previous PI approaches. In addition, we evaluate the robustness of MATE in more realistic scenarios, where agents can defect from the protocol and where communication failures can occur.

KEYWORDS

Multi-Agent Learning; Reinforcement Learning; Mutual Acknowledgments; Peer Incentivization; Emergent Cooperation

ACM Reference Format:
Thomy Phan, Felix Sommer, Philipp Altmann, Fabian Ritz, Lenz Belzner, and Claudia Linnhoff-Popien. 2022. Emergent Cooperation from Mutual Acknowledgment Exchange. In *Proc. of the 21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2022), Online, May 9–13, 2022*, IFAAMAS, 11 pages.

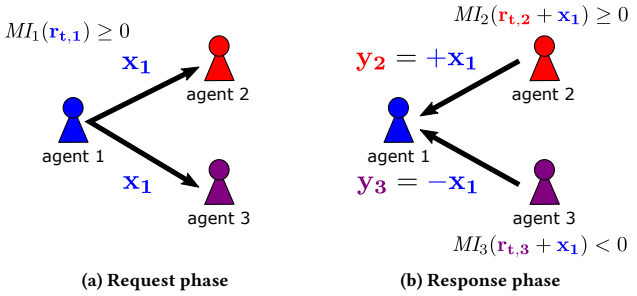
1 INTRODUCTION

Many potential AI scenarios like autonomous driving [30], smart grids [8], or general IoT scenarios [5], where multiple autonomous systems coexist within a shared environment, can be naturally modeled as self-interested *multi-agent system (MAS)* [4, 20]. In self-interested MAS, each autonomous system or agent attempts to achieve an individual goal while adapting to its environment, i.e., other agents' behavior [10]. Conflict and competition are common in such systems due to opposing goals or shared resources [20, 26].

In order to maximize social welfare or efficiency in such MAS, all agents need to cooperate which requires them to refrain from selfish and greedy behavior for the greater good. The tension between individual and collective rationality is typically modeled as *social dilemma (SD)* [28]. SDs can be extended to *sequential social dilemmas (SSD)* to model more realistic scenarios [17].

Multi-agent reinforcement learning (MARL) has become popular to model individually rational agents in SDs and SSDs to examine emergent behavior [4, 17, 24–26]. The goal of each agent is defined by an individual reward function. Non-cooperative game theory and empirical studies have shown that naive MARL approaches commonly fail to learn cooperative behavior due to individual selfishness and lacking benevolence towards other agents, which leads to defective behavior [1, 10, 17, 37].

One reason for mutual defection is *non-stationarity*, where naively learning agents do not consider the learning dynamics of other agents but only adapt reactively [4, 12, 16, 35]. This can cause agents to defect from mutual cooperation as studied extensively for the prisoner's dilemma [1, 10, 17, 28]. To mitigate this problem, some approaches propose to adapt the learning rate based on the outcome [3, 23, 40] or to integrate information of other agents' adaptation like gradients or opponent models [10, 15, 19]. These



(a) Request phase **(b) Response phase**

Figure 1: MATE protocol example. (a) If agent 1 estimates a monotonic improvement $MI_1(r_{t,1}) \geq 0$ of its situation, it "thanks" its neighbor agents 2 and 3 by sending an *acknowledgment request* $x_1 > 0$ as reward. (b) Agent 2 and 3 check if the request x_1 monotonically improves their own situation along with their own respective reward. If so, a positive reward (e.g., $y_2 = +x_1$) is sent back as a response. If not, a negative reward (e.g., $y_3 = -x_1$) is sent back.

Proc. of the 21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2022), P. Faliszewski, V. Mascardi, C. Pelachaud, M.E. Taylor (eds.), May 9–13, 2022, Online.

approaches are either tabular or require full observability in order to observe each other's behavior, thus do not scale to complex domains. Furthermore, some approaches require knowledge about other agents' objective to estimate their degree of adaptation which could violate privacy [10, 19].

Another reason for mutual defection is the *reward structure* which was found to be crucial for social intelligence [17, 31]. Prior work has shown that adequate reward formulations can lead to emergent cooperation in particular domains [2, 6, 7, 13, 27]. However, finding an appropriate reward formulation for any domain is generally not trivial. Recent approaches adapt the reward dynamically to drive all agents towards cooperation [14, 15, 42]. *Peer incentivization (PI)* is a distributed approach, where all agents learn to reward or to penalize each other which often leads to emergent cooperation [22, 29, 38, 42]. Current PI mechanisms implicitly assume a flawless communication channel in order to exchange rewards. These rewards are assumed to be simply integrated into the learning process without any chance to respond with feedback. Furthermore, most PI approaches rely on global information like joint actions [42], a central market function [29], or publicly available information [38] which limits scalability and applicability to real-world scenarios, where only local information is accessible.

In this paper, we propose *Mutual Acknowledgment Token Exchange (MATE)*, a PI approach defined by a two-phase communication protocol as shown in Fig. 1 to mutually exchange acknowledgment tokens to shape individual rewards. Each agent evaluates the monotonic improvement of its individual situation in order to accept or reject acknowledgment requests from other agents. MATE is completely decentralized and only requires local communication and information without knowing the objective of other agents or any public information. Our contributions include:

- The concept of monotonic improvement, where each agent locally evaluates its individual situation to estimate the reliability of the environment, i.e., other agents' behavior.
- The MATE communication protocol and reward formulation using monotonic improvement estimation.
- An empirical evaluation of MATE in three SD domains and a comparison with other PI approaches w.r.t. different metrics. Our results show that MATE is able to achieve and maintain significantly higher levels of cooperation than previous PI approaches. In addition, we evaluate the robustness of MATE in more realistic scenarios, where agents can defect from the protocol and where communication failures can occur.

2 BACKGROUND

2.1 Problem Formulation

We formulate self-interested MAS as partially observable *stochastic game* $M = \langle \mathcal{D}, \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \mathcal{Z}, \Omega \rangle$, where $\mathcal{D} = \{1, ..., N\}$ is a set of agents i, $\mathcal{S}$ is a set of states s_t at time step t, $\mathcal{A} = \langle \mathcal{A}_1, ..., \mathcal{A}_N \rangle = \langle \mathcal{A}_i \rangle_{i \in \mathcal{D}}$ is the set of joint actions $a_t = \langle a_{t,i} \rangle_{i \in \mathcal{D}}$, $\mathcal{P}(s_{t+1}|s_t, a_t)$ is the transition probability, $\langle r_{t,i} \rangle_{i \in \mathcal{D}} = \mathcal{R}(s_t, a_t) \in \mathbb{R}$ is the joint reward, $\mathcal{Z}$ is a set of local observations $z_{t,i}$ for each agent i, and $\Omega(s_t, a_t) = z_{t+1} = \langle z_{t+1,i} \rangle_{i \in \mathcal{D}} \in \mathcal{Z}^N$ is the subsequent joint observation. Each agent i maintains a local *history* $\tau_{t,i} \in (\mathcal{Z} \times \mathcal{A}_i)^t$. $\pi_i(a_{t,i}|\tau_{t,i})$ is the action selection probability represented by the individual *policy* of agent i. In addition, we assume each agent i to have a *neighborhood* $\mathcal{N}_{t,i} \subseteq \mathcal{D} - \{i\}$ of other agents at every time step t which is domain dependent as suggested in [43].

π_i is evaluated with a *value function* $V_i^{\pi}(s_t) = \mathbb{E}_{\pi}[G_{t,i}|s_t]$ for all $s_t \in \mathcal{S}$, where $G_{t,i} = \sum_{k=0}^{\infty} \gamma^k r_{t+k,i}$ is the individual and discounted *return* of agent $i \in \mathcal{D}$ with discount factor $\gamma \in [0, 1)$ and $\pi = \langle \pi_j \rangle_{j \in \mathcal{D}}$ is the *joint policy* of the MAS. The goal of agent i is to find a *best response* π_i^* with $V_i^* = max_{\pi_i} V_i^{\langle \pi_i, \pi_{-i} \rangle}$ for all $s_t \in \mathcal{S}$, where π_{-i} is the joint policy *without* agent i. In practice, the global state s_t is not directly observable for any agent i such that V_i is approximated with local information, i.e., $\tau_{t,i}$ instead [14, 17, 22, 26].

We define the *efficiency* of a MAS or *utilitarian metric (U)* by the sum of all individual rewards until time step T:

$$U = \sum_{i \in \mathcal{D}} R_i \tag{1}$$

where $R_i = \sum_{t=0}^{T-1} r_{t,i}$ is the *undiscounted return* or *sum of rewards* of agent i starting from start state s_0.

2.2 Multi-Agent Reinforcement Learning

We focus on decentralized or independent learning, where each agent i optimizes its policy π_i based on local information like $\tau_{t,i}$, $a_{t,i}$, $r_{t,i}$, $z_{t+1,i}$ (and optionally information obtained from its neighborhood $\mathcal{N}_{t,i}$) using *reinforcement learning (RL)* techniques, e.g., policy gradient methods as explained in Section 2.3 [10, 35, 43]. *Naive (independent) learning* introduces non-stationarity due to simultaneously adapting agents which continuously changes the environment dynamics [12, 16, 20]. This can cause naive learners to learn overly greedy and exploitative policies which defect from any cooperative behavior [10, 17].

2.3 Policy Gradient Reinforcement Learning

Policy gradient RL is a popular approach to approximate best responses π_i^* for each agent i [10, 21, 42]. A function approximator $\hat{\pi}_{i,\theta_i} \approx \pi_i^*$ with parameters θ_i is trained with gradient ascent on an estimate of $J = \mathbb{E}_{\pi}[G_{0,i}]$ [41]. Most policy gradient methods use gradients g of the following form [34]:

$$g = (G_{t,i} - b_i(s_t)) \nabla_{\theta_i} log \hat{\pi}_{i,\theta_i}(a_{t,i}|\tau_{t,i}) \tag{2}$$

where $b_i(s_t)$ is some state-dependent *baseline*. In practice, $b_i(s_t)$ is replaced by a value function approximation $\hat{V}_{i,\omega_i}(\tau_{t,i}) \approx V_i^{\hat{\pi}}(s_t)$ which is learned with parameter vector ω_i [10]. For simplicity, we omit the parameter indices θ_i, ω_i and write $\hat{\pi}_i$, $\hat{V}_i$ instead.

3 RELATED WORK

MARL is a long standing research field with rapid progress and success in challenging domains [4, 20, 35, 39]. Different studies have been conducted on various complex SSD domains, where interesting phenomena like group hunting, attacking and dodging, or flocking have been observed [17, 24–26]. Independent MARL like naive learning has been widely used in most studies to model agents with individual rationality [10, 35].

Non-stationarity of naive learning is one reason why agents fail to learn cooperative behavior in SDs [4, 12, 16, 20, 35]. To mitigate this issue, different learning rates can be used depending on the outcome [3, 23, 40]. Another approach is to integrate "opponent awareness" into the learning rule by using or approximating other

agents' gradients [10, 19]. For that, the objectives and histories of other agents' need to be known, thus requiring full observability. Furthermore, higher order derivatives (at least second order) are required which is computationally expensive for function approximators with many learnable parameters like deep neural networks.

PI approaches have been introduced recently to encourage cooperative behavior in a distributed fashion via additional rewards. Multi-agent *Gifting* has been proposed in [22], which extends the action space of each agent i with a gifting action to give a positive reward to other agents $j \in \mathcal{N}_{t,i}$. *Learning to Incentivize Other learning agents (LIO)* is a related approach, which learns an incentive function for each agent i that conditions on the joint action of all other agents $j \neq i$ (thus assuming full observability) in order to compute nonnegative incentive rewards for them [42]. Both Gifting and LIO are unidirectional PI approaches, where agents have neither the ability to respond nor to penalize each other.

[29] devised a market-based PI approach, where the action space is extended by joint market actions to enable bilateral agreements between agents. A central market function is required which redistributes rewards depending on selling-buying relationships. This approach is intractable for large and complex scenarios because of the exponential growth of the individual action space, since each agent has to additionally decide on a joint market action. Furthermore, this approach does not enable penalization of agents. Another approach based on public sanctioning has been proposed in [38]. Agents can reward or penalize each other which is made public to all other agents. Learning is conditioned on these public sanctioning events and agents can decide based on known group behavior patterns, whether to reward or to penalize other agents' behavior.

Direct reciprocity (DR) is an alternative approach to emergent cooperation in evolutionary settings [36]. Agents in a population can choose either to cooperate or defect based on previous interactions and the probability of future interactions. Thus, DR requires full observability of all other agents' actions within an interaction and a clear notion of cooperation and defection which can only be assumed for simple games [17, 26].

4 MUTUAL ACKNOWLEDGMENT TOKEN EXCHANGE (MATE)

We assume a decentralized MARL setting as formulated in Algorithm 1, where at every time step t each agent i with history $\tau_{t,i}$, policy approximation $\hat{\pi}_i$, and value function approximation $\hat{V}_i$ observes its neighborhood $\mathcal{N}_{t,i}$ and executes an action $a_{t,i} \sim \pi_i(a_{t,i}|\tau_{t,i})$ in state s_t. After all actions $a_t \in \mathcal{A}$ have been executed, the environment transitions into a new state $s_{t+1} \sim \mathcal{P}(s_{t+1}|s_t, a_t)$ which is observed by each agent i through reward $r_{t,i}$ and observation $z_{t+1,i}$. All agents collect their respective *experience tuple* $e_{t,i} = \langle \tau_{t,i}, a_{t,i}, r_{t,i}, z_{t+1,i} \rangle$ for PI [22, 29, 42] and independent adaptation of $\hat{\pi}_i$ and $\hat{V}_i$ [10, 17, 26].

4.1 Monotonic Improvement

After obtaining their respective experience tuples $e_{t,i}$, all agents can evaluate the *monotonic improvement* of their individual situation with a metric $MI_{e_{t,i},\hat{V}_i}$ or MI_i for short based on local information, i.e., rewards $r_{t,i}$, histories $\tau_{t,i}$, and messages exchanged with other agents $j \in \mathcal{N}_{t,i}$. Given some *arbitrary reward* $\hat{r}_{t,i}$, which could be either the original reward $r_{t,i}$ or some shaped reward, agent i can assume a monotonic improvement of its situation when $MI_i(\hat{r}_{t,i}) \geq 0$. Note that we consider the case of $MI_i(\hat{r}_{t,i}) = 0$ as monotonic improvement in particular to encourage agents to maintain their cooperative behavior instead of falling back to defective strategies.

MI_i represents a heuristic local reliability measure to predict if an agent i can rely on its environment represented by other agents $j \in \mathcal{N}_{t,i}$ without loosing performance. Since MI_i can be measured online, agent i is able to react accordingly at any time step by either encouraging other agents j if $MI_i(\hat{r}_{t,i}) \geq 0$ to reinforce its situation or by discouraging them if $MI_i(\hat{r}_{t,i}) < 0$.

In this paper, we regard a *reward-based* and a *temporal difference (TD)*-based approach to compute MI_i.

The reward-based approach computes $MI_i = MI_i^{rew}$ as follows:

$$MI_i^{rew}(\hat{r}_{t,i}) = \hat{r}_{t,i} - \overline{r_{t,i}} \tag{3}$$

where $\overline{r_{t,i}} = \frac{1}{t}\sum_{k=0}^{t-1} \hat{r}_{k,i}$ is the average of all (shaped) rewards before time step t. MI_i^{rew} estimates the expected *short-term* improvement of agent i, i.e., how $\hat{r}_{t,i}$ compares to all rewards obtained so far.

The TD-based approach computes $MI_i = MI_i^{TD}$ as follows:

$$MI_i^{TD}(\hat{r}_{t,i}) = \hat{r}_{t,i} + \gamma \hat{V}_i(\tau_{t+1,i}) - \hat{V}_i(\tau_{t,i}) \tag{4}$$

which corresponds to the TD residual w.r.t. to some *arbitrary reward* $\hat{r}_{t,i}$ and estimates the expected *long-term* improvement of agent i, i.e., how $\hat{r}_{t,i}$ and $\tau_{t+1,i}$ improve or degrade the situation of agent i w.r.t. future time steps [32, 33].

Note that both MI_i^{rew} and MI_i^{TD} only depend on local information like the reward $\hat{r}_{t,i}$, the value function $\hat{V}_i$, or the experience tuple $e_{t,i}$ and enable efficient online evaluation at every time step.

Algorithm 1 Multi-Agent Reinforcement Learning with MATE

Initialize parameters for $\hat{\pi}_i$ and $\hat{V}_i$ for all agents $i \in \mathcal{D}$.
for episode $m = 1, E$ **do**
  Sample s_0 and set $\tau_{0,i}$ for all agents $i \in \mathcal{D}$
  for time step $t = 0, T-1$ **do**
    for agent $i \in \mathcal{D}$ **do** ▷ Decision making in parallel
      Observe neighborhood $\mathcal{N}_{t,i}$
      $a_{t,i} \sim \hat{\pi}_i(a_{t,i}|\tau_{t,i})$
    $a_t \leftarrow \langle a_{t,i} \rangle_{i \in \mathcal{D}}$
    $s_{t+1} \sim \mathcal{P}(s_{t+1}|s_t, a_t)$
    $z_{t+1} \leftarrow \Omega(s_t, a_t)$
    $\langle r_{t,i} \rangle_{i \in \mathcal{D}} \leftarrow \mathcal{R}(s_t, a_t)$
    for agent $i \in \mathcal{D}$ **do** ▷ Communication in parallel
      $e_{t,i} \leftarrow \langle \tau_{t,i}, a_{t,i}, r_{t,i}, z_{t+1,i} \rangle$
      $\hat{r}_{t,i}^{MATE} \leftarrow MATE(MI_i, \hat{V}_i, \mathcal{N}_{t,i}, \tau_{t,i}, e_{t,i})$ (See Algorithm 2)
      $e_{t,i} \leftarrow \langle \tau_{t,i}, a_{t,i}, \hat{r}_{t,i}^{MATE}, z_{t+1,i} \rangle$
      Update $\tau_{t,i}$ to $\tau_{t+1,i}$ and store $e_{t,i}$
  for agent $i \in \mathcal{D}$ **do** ▷ Update in parallel
    Update $\hat{\pi}_i$ and $\hat{V}_i$ using all $e_{t,i}$ of episode m

4.2 MATE Protocol and Reward

MATE defines a two-phase communication protocol consisting of a *request phase* and a *response phase* as shown in Fig. 1.

In the request phase (Fig. 1a), each agent i checks its current situation with its original reward $r_{t,i}$. If $MI_i(r_{t,i}) \geq 0$, the agent sends a *token* $x_i = x_{token} > 0$ as an *acknowledgment request* to all other agents $j \in \mathcal{N}_{t,i}$ which can be interpreted as a reward. We assume all tokens to have a fixed value x_{token} which can be set specifically for particular domains. The request phase could be interpreted as an opportunity to "thank" other agents for supporting one's own monotonic improvement which is common practice in human society. Note that the fixed token value x_{token} does not directly reveal an agent's objective or value function.

In the response phase (Fig. 1b), all request receiving agents $j \in \mathcal{N}_{t,i}$ check if the request token x_i helps to monotonically improve their own situation along with their respective original reward $r_{t,j}$. If $MI_j(r_{t,j}+x_i) \geq 0$, then agent j accepts the request with a positive *response token* $y_j = +x_i$ which establishes a *mutual acknowledgment* between agent i and j for time step t. However if $MI_j(r_{t,j}+x_i) < 0$, then agent j rejects the request with a negative response token $y_j = -x_i$, because the received request token x_i is not sufficient to preserve or to compensate for the situation of agent j.

After both communication phases, the shaped reward $\hat{r}_{t,i}^{MATE}$ for each agent i is computed as follows:

$$\begin{aligned}\hat{r}_{t,i}^{MATE} &= r_{t,i} + \hat{r}_{req} + \hat{r}_{res} \\ &= r_{t,i} + max\{\langle x_j \rangle_{j \in \mathcal{N}_{t,i}}\} + min\{\langle y_j \rangle_{j \in \mathcal{N}_{t,i}}\}\end{aligned} \tag{5}$$

where $\hat{r}_{req} \in \{0, x_{token}\}$ is the aggregation of all received requests x_j and $\hat{r}_{res} \in \{-x_{token}, 0, x_{token}\}$ is the aggregation of all received responses y_j. When $\hat{r}_{req} + \hat{r}_{res} = 0$ for all time steps t, then agent i would adapt like a naive learner. Although $\hat{r}_{req}$ and $\hat{r}_{res}$ could be formulated as sums over all requests or responses respectively, we prefer *max* and *min* aggregation to prevent single neighbor agents to be "voted out" by all other agents in $\mathcal{N}_{t,i}$, thus pushing the interaction towards overall cooperation in a completely decentralized way. Furthermore, the *max* and *min* operators keep the reward $\hat{r}_{t,i}^{MATE}$ bounded within $[r_{t,i} - x_{token}, r_{t,i} + 2x_{token}]$ which can alleviate undesired exploitation of the PI mechanism, e.g., by becoming "lazy" to avoid harming other agents while getting rewarded or by deviating from the protocol such that only positive rewards are used for learning, e.g., by ignoring responses.

The complete formulation of MATE at time step t for any agent i is given in Algorithm 2. MI_i is a metric for estimating the individual monotonic improvement, $\hat{V}_i$ is the approximated value function, $\mathcal{N}_{t,i}$ is the current neighborhood, $\tau_{t,i}$ is the history, and $e_{t,i}$ is the experience tuple obtained at time step t. MATE computes and returns the shaped reward $\hat{r}_{t,i}^{MATE}$ (Eq. 5), which can be used to update $\hat{\pi}_i$ and $\hat{V}_i$ according to line 18 in Algorithm 1.

4.3 Discussion of MATE

MATE aims at incentivizing all agents to learn cooperative behavior with a decentralized two-phase communication protocol. Like Gifting [22], we focus on *fixed rewards* to simplify evaluation and to focus on the main aspects of our approach. If x_{token} is smaller than the highest positive reward, then agents might not be sufficiently incentivized for cooperation. However, if x_{token} significantly exceeds the highest domain penalty, then single agents may learn to "bribe" all other agents, thus leading to imbalance. An adaptation of x_{token} to more flexible values like in LIO [42] is left for future work. In contrast to Gifting and LIO, MATE enables penalization of other agents by explicitly rejecting acknowledgment requests, which has a negative effect on the requesting agent's reward, i.e., the response term $\hat{r}_{res} = min\{\langle y_j \rangle_{j \in \mathcal{N}_{t,i}}\}$ in Eq. 5.

Algorithm 2 Mutual Acknowledgment Token Exchange (MATE)

1: **procedure** $MATE(MI_i, \hat{V}_i, \mathcal{N}_{t,i}, \tau_{t,i}, e_{t,i})$
2: $\quad\hat{r}_{req} \leftarrow 0, \hat{r}_{res} \leftarrow 0$
3: $\quad$**if** $MI_i(r_{t,i}) \geq 0$ **then**
4: $\quad\quad$Send acknowledgment request $x_i = x_{token}$ to all $j \in \mathcal{N}_{t,i}$
5: $\quad$**for** neighbor agent $j \in \mathcal{N}_{t,i}$ **do** ▷ Respond to requests
6: $\quad\quad$**if** request x_j received from j **then**
7: $\quad\quad\quad\hat{r}_{req} \leftarrow max\{\hat{r}_{req}, x_j\}$
8: $\quad\quad\quad$**if** $MI_i(r_{t,i} + x_j) \geq 0$ **then**
9: $\quad\quad\quad\quad$Send response $y_i = +x_j$ to agent j
10: $\quad\quad\quad$**else**
11: $\quad\quad\quad\quad$Send response $y_i = -x_j$ to agent j
12: $\quad$**if** $MI_i(r_{t,i}) \geq 0$ **then** ▷ If requests have been sent before
13: $\quad\quad$**for** neighbor agent $j \in \mathcal{N}_{t,i}$ **do** ▷ Receive responses
14: $\quad\quad\quad$**if** response y_j received from j **then**
15: $\quad\quad\quad\quad\hat{r}_{res} \leftarrow min\{\hat{r}_{res}, y_j\}$
$\quad$**return** $r_{t,i} + \hat{r}_{req} + \hat{r}_{res}$ ($\hat{r}_{t,i}^{MATE}$ as defined in Eq. 5)

MATE agents completely rely on *local information*, i.e., their own value function approximation $\hat{V}_i$, their own experience tuples $e_{t,i}$, and messages exchanged within their local neighborhood $\mathcal{N}_{t,i}$ and do not require knowledge about other agent's objectives, or central instances like market functions or public information as suggested in [10, 19, 21, 29, 38].

5 EXPERIMENTAL SETUP

5.1 Evaluation Domains

We implemented[1] three SD domains based on previous work [10, 22, 26]. At every time step, the order of agent actions is randomized to resolve conflicts (e.g., when multiple agents step on a coin or tag each other simultaneously). For all domains, we measure the degree of cooperation by the efficiency (U) according to Eq. 1.

Iterated Prisoner's Dilemma (IPD). *IPD* is an iterated version of the 2-player Prisoner's Dilemma with payoff table shown in Fig. 3a. Both agents observe the previous joint action $z_{t,i} = a_{t-1}$ at every time step t, which is the zero vector at start state s_0. One nash equilibrium is to always defect (DD) with an average efficiency of

[1] Code available at https://github.com/thomyphan/emergent-cooperation.

$U = -2 - 2 = -4$ per time step. Cooperative policies are able to achieve higher efficiency up to $U = -1 - 1 = -2$ per time step. An episode consists of 150 iterations and we set $\gamma = 0.95$. The neighborhood $\mathcal{N}_{t,i} = \{j\}$ is defined by the other agent $j \neq i$.

Coin. *Coin[N]* is an SSD as shown in Fig. 2a and consists of $N \in \{2, 4\}$ agents with different colors, which start at random positions and have to collect a coin with a random color and a random position [10, 18]. If an agent collects a coin, it receives a reward of +1. However, if the coin has a different color than the collecting agent, another agent with the actual matching color is penalized with -2. After being collected, the coin respawns randomly with a new random color. All agents can observe the whole field and are able to move north, south, west, and east. An agent is only able to determine if a coin has the same or a different color than itself, but it is unable to distinguish anything further between colors. An episode terminates after 150 time steps and we set $\gamma = 0.95$. The neighborhood $\mathcal{N}_{t,i} = \mathcal{D} - \{i\}$ is defined by all other agents $j \neq i$. In addition to the efficiency, we measure the *"own coin" rate* $P(\textit{own coin}) = \frac{\textit{\# collected coins with same color}}{\textit{\# all collected coins}}$ based on the coins collected by each agent.

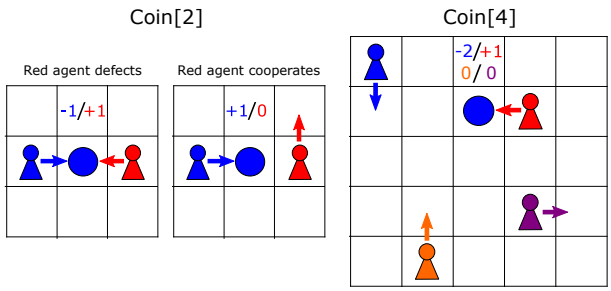
Harvest. *Harvest[N]* is an SSD as shown in Fig. 2b and consists of $N \in \{6, 12\}$ agents (red circles), which start at random positions and have to collect apples (green squares). The apple regrowth rate as depends on the number of surrounding apples, where more neighbor apples lead to a higher regrowth rate [26]. If all apples are harvested, then no apple will grow anymore until the episode terminates. At every time step, all agents receive a time penalty of -0.01. For each collected apple, an agent receives a reward of +1. All agents have a 7×7 field of view and are able to do nothing, move north, south, west, east, and tag other agents within their view with a tag beam of width 5 pointed to a specific cardinal direction. If an agent is tagged, it is unable to act for 25 time steps. Tagging does not directly penalize the tagged agents nor reward the tagging agent. An episode terminates after 250 time steps and we set $\gamma = 0.99$. The neighborhood $\mathcal{N}_{t,i}$ is defined by all other agents $j \neq i$ being in sight of i. In addition to the efficiency, we measure *equality (E)* (1 minus the Gini coefficient), *sustainability (S)* (the average time at which apples are collected), and *peace (P)* (the average number of untagged agents at any time step) [26] to analyze the degree of cooperation in more detail.

5.2 MARL algorithms

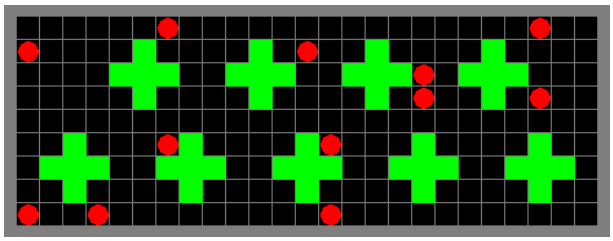
We implemented MATE as specified in Algorithm 2 with MI_i^{TD} (Eq. 4) and MI_i^{rew} (Eq. 3), which we refer to as *MATE-TD* and *MATE-rew* respectively and always set $x_{token} = 1$. Our base algorithm is an *independent actor-critic* to approximate $\hat{\pi}_i$ and $\hat{V}_i$ for each agent i according to Eq. 2, which we refer to as *Naive Learning* [10].

In addition, we implemented *LIO* [42], the zero-sum and replenishable budget version of *Gifting* [22], and a *Random* baseline.

Due to the high computational demand of *LOLA-PG*, which requires the computation of the second order derivative for deep neural networks, we directly include the performance as reported in the paper [10] in *IPD* and *Coin[2]* for comparison.



(a) Coin



(b) Harvest (layout used for $N = 6$ and $N = 12$)

Figure 2: SSD environments for evaluation: (a) In Coin, each agent gets a reward of +1 when collecting a coin. However, other agents can be penalized with -2 when the collected coin does not match with the collecting agent's color. (b) In Harvest, all agents (red circles) need to collect apples (green squares) while avoiding to be tagged and exhaustion of all apples which would prevent regrowth of apples.

5.3 Neural Network Architectures

We implemented $\hat{\pi}_i$ and $\hat{V}_i$ for each agent i as multilayer perceptron (MLP). Since *Coin[N]* and *Harvest[N]* are gridworlds, states and observations are encoded as multi-channel image as proposed in [11, 17]. The observations of *IPD* are the vector-encoded joint actions of the previous time step [10]. The multi-channel images of *Coin[N]* and *Harvest[N]* were flattened, before being fed into the networks. The output of $\hat{\pi}_i$ has $|\mathcal{A}_i|$ ($|\mathcal{A}_i| + 1$ for *Gifting*) units with softmax activation. The output of $\hat{V}_i$ consists of a single linear unit. The incentive function of *LIO* has a similar architecture with the joint action a_t (excluding $a_{t,i}$) concatenated with the flattened observations as input and $N - 1$ output units with sigmoid activation. The hyperparameters and architecture information are listed in Table 1 and further details are in the appendix.

6 RESULTS

For each experiment all respective algorithms were run 20 times to report the average metrics and the 95% confidence interval. The *Random* baseline was run 1000 times to estimate its expected performance for each domain.

6.1 Performance Evaluation

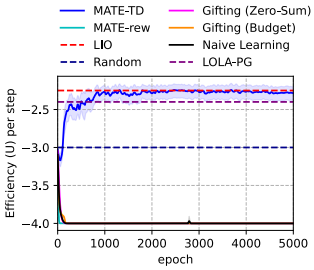
The results for IPD are shown in Fig. 3b. *MATE-TD*, *LIO*, and *LOLA-PG* achieve the highest average efficiency per step. Both *Gifting*

hyperparameter	value
x_{token}	1
discount factor γ	0.95 (0.99 for *Harvest[N]*)
batch size	10 episodes
optimizer	ADAM
learning rate	0.001
clip norm	1
number of hidden layers	2
units per hidden layer	64
hidden layer activation	ELU

Table 1: Hyperparameters of MATE and the neural networks.

	C	D
C	$(-1, -1)$	$(-3, 0)$
D	$(0, -3)$	$(-2, -2)$

(a) Prisoner's Dilemma payoffs



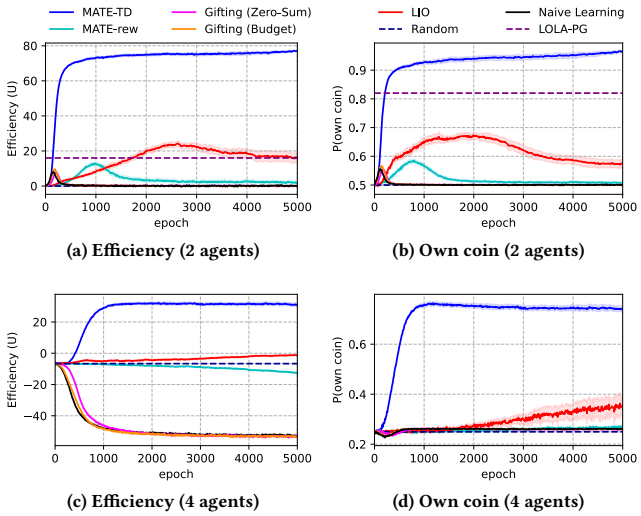
(b) Efficiency in *IPD*

Figure 3: (a) Payoff matrix used in IPD (b) Learning progress of MATE variants, Gifting variants, Naive Learning, and Random in *IPD*. The results of LIO and LOLA-PG are from the respective papers.

variants, *Naive Learning*, and *MATE-rew* converge to mutual defection, which is significantly less efficient than *Random*.

The results for *Coin[2]* and *Coin[4]* are shown in Fig. 4. In both scenarios, *MATE-TD* is the significantly most efficient approach with the highest "own coin" rate. *LIO* is the second most efficient approach in both scenarios. In *Coin[2]*, *LIO*'s efficiency first surpasses *LOLA-PG* and then decreases to a similar level. However, the "own coin" rate of *LOLA-PG* is higher, which indicates that one *LIO* agent mostly collects all coins while incentivizing the other respective agent to move elsewhere. In *Coin[4]*, *LIO* is more efficient than *Random* and achieves a slightly higher "own coin" rate than the other PI baselines. *MATE-rew* is the fourth most efficient in *Coin[2]* (after *LOLA-PG* and *LIO*) and *Coin[4]* (after *Random*), but its "own coin" rate is similar to *Random*. Both *Gifting* variants and *Naive Learning* perform similarly to *Random* in *Coin[2]* but are significantly less efficient than *Random* in *Coin[4]*.

The results for *Harvest[6]* and *Harvest[12]* are shown in Fig. 5. All MARL approaches are more efficient, sustainable, and peaceful than *Random*. In *Harvest[6]*, *MATE-TD*, *LIO*, both *Gifting* variants, and *Naive Learning* are similarly efficient and sustainable with similar equality, while *MATE-TD* achieves slightly more peace than all other baselines. In *Harvest[12]*, *MATE-TD* achieves the highest efficiency, equality, and sustainability over time while being second most peaceful after *MATE-rew*, which achieves the highest and most stable level of peace. Both *Gifting* variants are slightly more efficient, sustainable, and peaceful than *Naive Learning* in *Harvest[12]*, while



(a) Efficiency (2 agents) **(b) Own coin (2 agents)**

(c) Efficiency (4 agents) **(d) Own coin (4 agents)**

Figure 4: Learning progress of MATE variants, LIO, Gifting variants, Naive Learning, and Random in Coin. The results of LOLA-PG are from the paper.

LIO is progressing slowlier than *Gifting* and *Naive Learning*, but eventually surpasses them w.r.t. efficiency, sustainability, and peace. *MATE-rew* is the least efficient and sustainable MARL approach which exhibits less significantly equality than *Random*. *LIO*, both *Gifting* variants, and *Naive Learning* first improve w.r.t. to all metrics but then exhibit a gradual decrease, which indicates that agents become more aggressive and tag each other in order to harvest all apples alone, which is known as *tragedy of the commons* [22, 26]. However, *MATE-TD* remains stable w.r.t. efficiency, equality, and sustainability in *Harvest[12]*.

6.2 Robustness against Protocol Defections

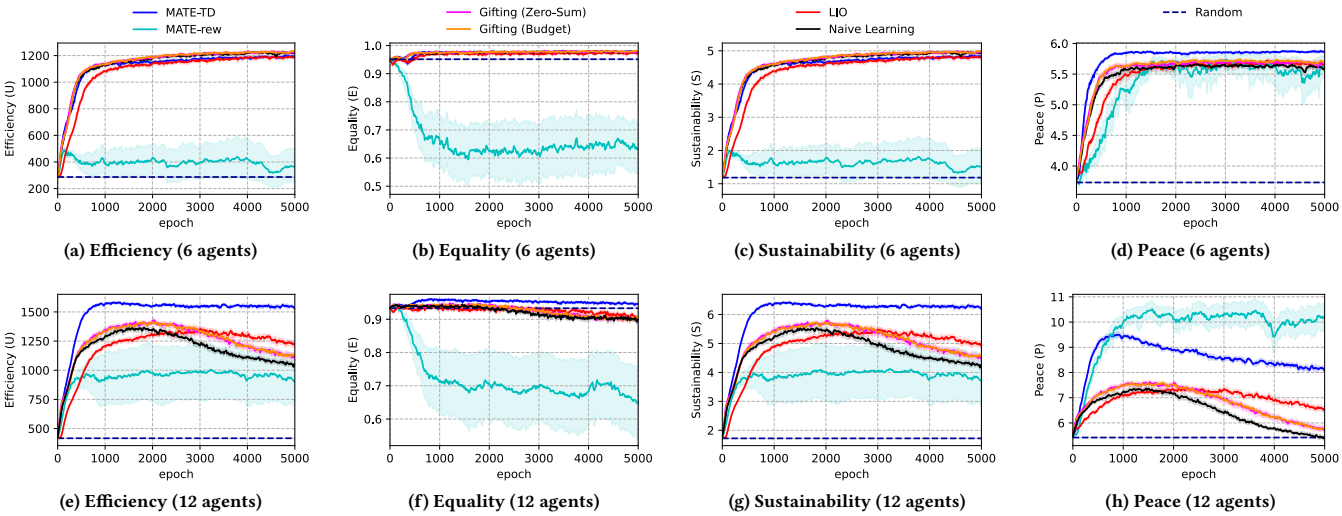
To evaluate robustness of *MATE-TD* against protocol defections, we introduce a single defective agent or *defector* $f \in \mathcal{D}$ which deviates from the communication protocol defined in Algorithm 2 and Fig. 1 in one of the following ways:

Complete The defector becomes a naive independent learner which does not participate in the communication rounds by skipping line 14 and 15 in Algorithm 1. Thus, the defector f simply learns with its original reward $r_{t,f}$.

Request The defector f does not send any acknowledgment requests by skipping line 4 in Algorithm 2 and receives no responses in return. However, requests from other agents $j \in \mathcal{N}_{t,f}$ can still be received. Thus, the defector's reward is defined by $\hat{r}^{MATE}_{t,f} = r_{t,f} + \hat{r}_{req} = r_{t,f} + max\{\langle x_j \rangle_{j \in \mathcal{N}_{t,f}}\}$.

Response The defector f can send acknowledgment requests but ignores all responses by skipping line 13-15 in Algorithm 2. In addition, it can receive requests from other agents $j \in \mathcal{N}_{t,f}$. Thus, the defector's reward $\hat{r}^{MATE}_{t,f}$ is the same as in the *Request* case above.

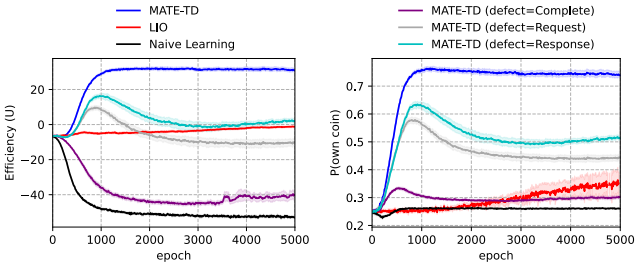
Note that we focus on variants that avoid penalization by other agents through the response term $\hat{r}_{res} = min\{\langle y_j \rangle_{j \in \mathcal{N}_{t,i}}\}$ of Eq. 5.



(a) Efficiency (6 agents) (b) Equality (6 agents) (c) Sustainability (6 agents) (d) Peace (6 agents)

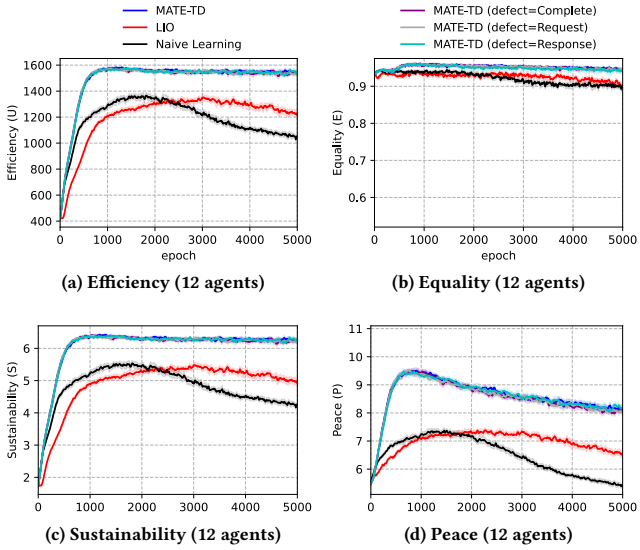
(e) Efficiency (12 agents) (f) Equality (12 agents) (g) Sustainability (12 agents) (h) Peace (12 agents)

Figure 5: Learning progress of MATE variants, LIO, Gifting variants, Naive Learning, and Random in Harvest.



(a) Efficiency (4 agents) (b) Own coin (4 agents)

Figure 6: Performance of MATE, defective MATE variants, LIO, and Naive Learning in Coin.



(a) Efficiency (12 agents) (b) Equality (12 agents)

(c) Sustainability (12 agents) (d) Peace (12 agents)

Figure 7: Performance of MATE, defective MATE variants, LIO, and Naive Learning in Harvest.

In our experiments, we use the notation *MATE-TD (defect=X)* for the inclusion of a defector f using a protocol defection strategy $X \in \{Complete, Request, Response\}$ as explained above.

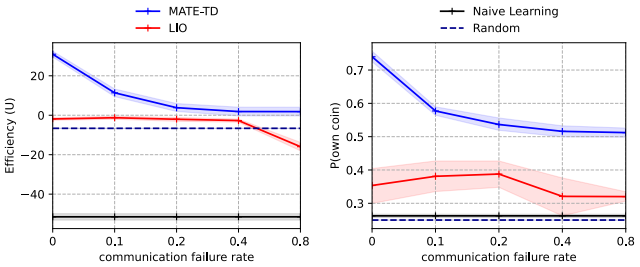
The results for *Coin[4]* are shown in Fig. 6. All defective *MATE-TD* variants are less efficient than *MATE-TD* but still more efficient with a higher "own coin" rate than *Naive Learning*. *MATE-TD (defect=Complete)* exhibits the least degree of cooperation. *MATE-TD (defect=Response)* is slightly more efficient than *LIO* and achieves a higher "own coin" rate. *MATE-TD (defect=Request)* is less efficient than *LIO* but its "own coin" rate is higher indicating that agents tend to refrain from collecting other agents' coins rather than greedily collecting them.

The results for *Harvest[12]* are shown in Fig. 7. All defective *MATE-TD* variants perform similarly to *MATE-TD* without any loss.

6.3 Robustness against Communication Failures

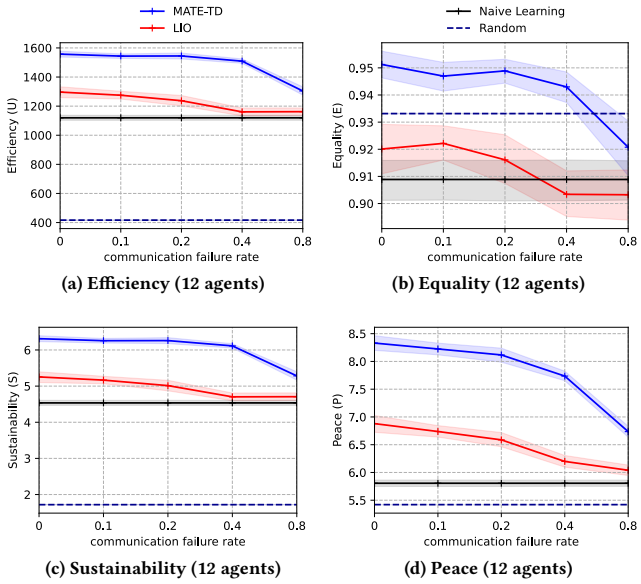
To evaluate robustness against communication failures, we introduce a *failure rate* $\delta \in [0, 1)$ specifying that an agent can fail to send or receive a message with a probability of δ. E.g., in the request phase in Fig. 1a, agent 1 could fail to send any request by skipping line 4 in Algorithm 2 with a probability of δ. If the requests are sent successfully, agent 2 or 3 can still fail at receiving agent 1's request by skipping lines 6-11 in Algorithm 2 with a probability of δ. The response phase in Fig. 1b is modeled analogously.

We evaluate the final training run performance of *MATE-TD* and *LIO* w.r.t. communication failure rates of $\delta \in \{0, 0.1, 0.2, 0.4, 0.8\}$ in *Coin[4]* and *Harvest[12]*. According to the corresponding neighborhood definitions in Section 5.1, communication in *Coin[4]* is *global*, where all-to-all communication is possible, while communication in *Harvest[12]* is *local* for *MATE-TD*, where all agents can



(a) Efficiency (4 agents) (b) Own coin (4 agents)

Figure 8: Performance of MATE, LIO, Naive Learning, and Random in Coin after 5000 epochs w.r.t. different communication failure rates.



(a) Efficiency (12 agents) (b) Equality (12 agents)

(c) Sustainability (12 agents) (d) Peace (12 agents)

Figure 9: Performance of MATE, LIO, Naive Learning, and Random in Harvest after 5000 epochs w.r.t. different communication failure rates.

only communicate with neighbor agents that are in their respective 7×7 field of view. *LIO* always uses global communication due to its incentive function formulation [42]. In addition, we compare with *Naive Learning* and *Random* as non-communicating baselines.

The results for *Coin[4]* are shown in Fig. 8. *MATE-TD* and *LIO* remain more efficient and cooperative than *Naive Learning* despite both approaches loosing performance with increasing δ. The average efficiency of *MATE-TD* is always nonnegative, while the efficiency of *LIO* decreases below the level of *Random*, when $\delta = 0.8$. The average "own coin" rate of *MATE-TD* is always at least 0.5, while the average "own coin" rate of *LIO* has a high variance ranging from 0.3 to 0.4. However, when $\delta = 0.8$, the average "own coin" rate of *LIO* is slightly above 0.3 with significantly less variance, while still being higher than the rates of *Naive Learning* and *Random*.

The results for *Harvest[12]* are shown in Fig. 9. The performance of *MATE-TD* is relatively robust for $\delta \geq 0.4$ but significantly drops when $\delta = 0.8$. However, *MATE-TD* still achieves the highest degree of cooperation w.r.t. all metrics except equality which gets worse than *Random* when $\delta = 0.8$. The cooperation level of *LIO* decreases slightly w.r.t. δ and is higher than *Random* except for equality which even falls below the level of *Naive Learning* when $\delta \leq 0.4$.

7 DISCUSSION

We presented MATE, a PI approach defined by a two-phase communication protocol to mutually exchange acknowledgment tokens to shape individual rewards. Each agent evaluates the monotonic improvement of its individual situation in order to accept or reject acknowledgment requests from other agents. MATE is completely decentralized and only requires local communication and information without knowing the objective of other agents or any public information. In addition to rewarding other agents, MATE enables penalization by explicitly rejecting acknowledgment requests, which has a negative effect on the requesting agent's reward, i.e., the minimization term in Eq. 5.

Our results show that MATE is able to achieve and maintain significantly higher levels of cooperation than previous PI approaches in SSDs like *Coin[2]*, *Coin[4]*, and *Harvest[12]*. Especially *Harvest[12]* emphasizes the capability of MATE to encourage stable cooperation despite the increased social pressure compared to *Harvest[6]*, where all alternative PI approaches easily learn to cooperate.

Estimating the monotonic short-term improvement via MI_i^{rew} (Eq. 3) can be beneficial compared to random acting and to some extent to naive learning in Coin (Fig. 4). However, considering the monotonic long-term improvement via MI_i^{TD} (Eq. 4) leads to significantly higher efficiency and cooperation w.r.t. various metrics in all domains, except peace in *Harvest[12]*. MATE with MI_i^{TD} is able to maintain cooperative behavior, in contrast to other approaches which become unstable and fall back to more defective strategies as observed in *Coin[2]*, *Coin[4]*, and *Harvest[12]* (Fig. 4 and 5).

MATE is suitable for more realistic scenarios, e.g., in adhoc teamwork or IoT settings, where single agents can deviate from the protocol, e.g., due to malfunctioning or selfishness, and where communication is not perfectly reliable: MATE is not affected by single protocol defectors in *Harvest[12]*, while its cooperation level decreases significantly in *Coin[4]*, where any deviation from the protocol can affect the whole MAS (Fig. 6 and 7). The protocol defection in *Coin[4]* emphasizes the importance of appropriate penalization mechanisms as proposed in our reward formulation in Eq. 5. MATE shows some robustness against communication failures in Fig. 8 and 9, where it is able to maintain its superior cooperation level even when communication fails with a probability of 80%. The difference in cooperation compared to LIO is especially evident in *Harvest[12]*, where MATE only uses local communication w.r.t. the agents' local neighborhoods $\mathcal{N}_{t,i}$. In this case, local failures with a rate of $\delta \leq 40\%$ do not affect the whole MAS, in contrast to *Coin[4]*, where the cooperation level already drops when $\delta \geq 10\%$.

Future work includes a combination of LIO and MATE in order to exchange tokens of different values and an integration of emergent communication techniques to create more adaptive and intelligent agents with social capabilities [9, 31].

REFERENCES

[1] Robert Axelrod and William Donald Hamilton. 1981. The Evolution of Cooperation. *science* 211, 4489 (1981), 1390–1396.

[2] Monica Babes, Enrique Munoz de Cote, and Michael L Littman. 2008. Social Reward Shaping in the Prisoner's Dilemma. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 3*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 1389–1392.

[3] Michael Bowling and Manuela Veloso. 2002. Multiagent Learning using a Variable Learning Rate. *Artificial Intelligence* 136, 2 (2002), 215–250.

[4] Lucian Buşoniu, Robert Babuška, and Bart De Schutter. 2008. Multi-Agent Reinforcement Learning: An Overview. *IEEE Transactions on Systems, Man, and Cybernetics—Part C: Applications and Reviews* 38, 2 (2008), 156–172.

[5] Shuiguang Deng, Zhengzhe Xiang, Peng Zhao, Javid Taheri, Honghao Gao, Jianwei Yin, and Albert Y Zomaya. 2020. Dynamical Resource Allocation in Edge for Trustable Internet-of-Things Systems: A Reinforcement Learning Method. *IEEE Transactions on Industrial Informatics* 16, 9 (2020), 6103–6113.

[6] Sam Devlin and Daniel Kudenko. 2011. Theoretical Considerations of Potential-based Reward Shaping for Multi-Agent Systems. In *The 10th International Conference on Autonomous Agents and Multiagent Systems*. ACM, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 225–232.

[7] Sam Devlin, Logan Yliniemi, Daniel Kudenko, and Kagan Tumer. 2014. Potential-based Difference Rewards for Multiagent Reinforcement Learning. In *Proceedings of the 2014 International Conference on Autonomous Agents and Multi-Agent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 165–172.

[8] AL Dimeas and ND Hatziargyriou. 2010. Multi-Agent Reinforcement Learning for Microgrids. In *IEEE PES General Meeting*. IEEE, 1–8.

[9] Jakob Foerster, Ioannis Alexandros Assael, Nando de Freitas, and Shimon Whiteson. 2016. Learning to Communicate with Deep Multi-Agent Reinforcement Learning. In *Advances in Neural Information Processing Systems*. Curran Associates Inc., Red Hook, NY, USA, 2137–2145.

[10] Jakob Foerster, Richard Y Chen, Maruan Al-Shedivat, Shimon Whiteson, Pieter Abbeel, and Igor Mordatch. 2018. Learning with Opponent-Learning Awareness. In *Proceedings of the 17th International Conference on Autonomous Agents and Multiagent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 122–130.

[11] Jayesh K Gupta, Maxim Egorov, and Mykel Kochenderfer. 2017. Cooperative Multi-Agent Control using Deep Reinforcement Learning. *Autonomous Agents and Multiagent Systems* 10642 (2017), 66–83.

[12] Pablo Hernandez-Leal, Michael Kaisers, Tim Baarslag, and Enrique Munoz de Cote. 2017. A Survey of Learning in Multiagent Environments: Dealing with Non-Stationarity. *arXiv preprint arXiv:1707.09183* (2017).

[13] Edward Hughes, Joel Z Leibo, Matthew Phillips, Karl Tuyls, Edgar Dueñez-Guzman, Antonio García Castañeda, Iain Dunning, Tina Zhu, Kevin McKee, Raphael Koster, et al. 2018. Inequity Aversion Improves Cooperation in Intertemporal Social Dilemmas. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*. Curran Associates Inc., Red Hook, NY, USA, 3330–3340.

[14] Max Jaderberg, Wojciech M Czarnecki, Iain Dunning, Luke Marris, Guy Lever, Antonio Garcia Castaneda, Charles Beattie, Neil C Rabinowitz, Ari S Morcos, Avraham Ruderman, et al. 2019. Human-Level Performance in 3D Multiplayer Games with Population-based Reinforcement Learning. *Science* 364, 6443 (2019), 859–865.

[15] Natasha Jaques, Angeliki Lazaridou, Edward Hughes, Caglar Gulcehre, Pedro Ortega, DJ Strouse, Joel Z Leibo, and Nando De Freitas. 2019. Social Influence as Intrinsic Motivation for Multi-Agent Deep Reinforcement Learning. In *Proceedings of the 36th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, 3040–3049.

[16] Guillaume J Laurent, Laëtitia Matignon, Le Fort-Piat, et al. 2011. The world of independent learners is not Markovian. *International Journal of Knowledge-based and Intelligent Engineering Systems* 15, 1 (2011), 55–64.

[17] Joel Z Leibo, Vinicius Zambaldi, Marc Lanctot, Janusz Marecki, and Thore Graepel. 2017. Multi-Agent Reinforcement Learning in Sequential Social Dilemmas. In *Proceedings of the 16th Conference on Autonomous Agents and Multiagent Systems (AAMAS '17)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 464–473.

[18] Adam Lerer and Alexander Peysakhovich. 2017. Maintaining Cooperation in Complex Social Dilemmas using Deep Reinforcement Learning. *arXiv preprint arXiv:1707.01068* (2017).

[19] Alistair Letcher, Jakob Foerster, David Balduzzi, Tim Rocktäschel, and Shimon Whiteson. 2019. Stable Opponent Shaping in Differentiable Games. *International Conference on Learning Representations* (2019).

[20] Michael L Littman. 1994. Markov Games as a Framework for Multi-Agent Reinforcement Learning. In *Machine Learning Proceedings 1994*. Morgan Kaufmann, San Francisco (CA), 157–163.

[21] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. 2017. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. In *Advances in Neural Information Processing Systems*, Vol. 30. Curran Associates, Inc.

[22] Andrei Lupu and Doina Precup. 2020. Gifting in Multi-Agent Reinforcement Learning. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 789–797.

[23] Laëtitia Matignon, Guillaume J Laurent, and Nadine Le Fort-Piat. 2007. Hysteretic Q-Learning: An Algorithm for Decentralized Reinforcement Learning in Cooperative Multi-Agent Teams. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 64–69.

[24] MIT Press 2019. *Emergent Escape-Based Flocking Behavior using Multi-Agent Reinforcement Learning*. ALIFE 2021: The 2021 Conference on Artificial Life, Vol. ALIFE 2019: The 2019 Conference on Artificial Life. MIT Press.

[25] MIT Press 2021. *A Sustainable Ecosystem through Emergent Cooperation in Multi-Agent Reinforcement Learning*. ALIFE 2021: The 2021 Conference on Artificial Life, Vol. ALIFE 2021: The 2021 Conference on Artificial Life. MIT Press.

[26] Julien Perolat, Joel Z. Leibo, Vinicius Zambaldi, Charles Beattie, Karl Tuyls, and Thore Graepel. 2017. A Multi-Agent Reinforcement Learning Model of Common-Pool Resource Appropriation. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Long Beach, California, USA) *(NIPS'17)*. Curran Associates Inc., Red Hook, NY, USA, 3646–3655.

[27] Alexander Peysakhovich and Adam Lerer. 2018. Prosocial Learning Agents Solve Generalized Stag Hunts Better than Selfish Ones. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS '18)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 2043–2044.

[28] Anatol Rapoport. 1974. Prisoner's Dilemma — Recollections and Observations. In *Game Theory as a Theory of a Conflict Resolution*. Springer, 17–34.

[29] Kyrill Schmid, Lenz Belzner, Robert Müller, Johannes Tochtermann, and Claudia Linnhoff-Popien. 2021. Stochastic Market Games. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, Zhi-Hua Zhou (Ed.). International Joint Conferences on Artificial Intelligence Organization, 384–390.

[30] Shai Shalev-Shwartz, Shaked Shammah, and Amnon Shashua. 2016. Safe Multi-Agent Reinforcement Learning for Autonomous Driving. *arXiv preprint arXiv:1610.03295* (2016).

[31] David Silver, Satinder Singh, Doina Precup, and Richard S Sutton. 2021. Reward is Enough. *Artificial Intelligence* 299 (2021), 103535.

[32] Richard S Sutton. 1988. Learning to Predict by the Methods of Temporal Differences. *Machine Learning* 3, 1 (1988), 9–44.

[33] Richard S Sutton and Andrew G Barto. 2018. *Reinforcement Learning: An Introduction*. MIT press.

[34] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. 2000. Policy Gradient Methods for Reinforcement Learning with Function Approximation. In *Advances in Neural Information Processing Systems*, S. Solla, T. Leen, and K. Müller (Eds.), Vol. 12. MIT Press, 1057–1063.

[35] Ming Tan. 1993. Multi-Agent Reinforcement Learning: Independent versus Cooperative Agents. In *Proceedings of the Tenth International Conference on International Conference on Machine Learning*. Morgan Kaufmann Publishers Inc., 330–337.

[36] Robert L Trivers. 1971. The Evolution of Reciprocal Altruism. *The Quarterly Review of Biology* 46, 1 (1971), 35–57.

[37] Paul AM Van Lange, Jeff Joireman, Craig D Parks, and Eric Van Dijk. 2013. The Psychology of Social Dilemmas: A Review. *Organizational Behavior and Human Decision Processes* 120, 2 (2013), 125–141.

[38] Eugene Vinitsky, Raphael Köster, John P Agapiou, Edgar Duéñez-Guzmán, Alexander Sasha Vezhnevets, and Joel Z Leibo. 2021. A Learning Agent that Acquires Social Norms from Public Sanctions in Decentralized Multi-Agent Settings. *arXiv preprint arXiv:2106.09012* (2021).

[39] Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. 2019. Grandmaster Level in StarCraft II using Multi-Agent Reinforcement Learning. *Nature* (2019), 1–5.

[40] Ermo Wei and Sean Luke. 2016. Lenient Learning in Independent-Learner Stochastic Cooperative Games. *The Journal of Machine Learning Research* 17, 1 (2016), 2914–2955.

[41] Ronald J Williams. 1992. Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning. *Machine learning* 8, 3 (1992), 229–256.

[42] Jiachen Yang, Ang Li, Mehrdad Farajtabar, Peter Sunehag, Edward Hughes, and Hongyuan Zha. 2020. Learning to Incentivize Other Learning Agents. *Advances in Neural Information Processing Systems* 33 (2020).

[43] Y Yang, R Luo, M Li, M Zhou, W Zhang, and J Wang. 2018. Mean Field Multi-Agent Reinforcement Learning. In *35th International Conference on Machine Learning, ICML 2018*, Vol. 80. PMLR, 5571–5580.

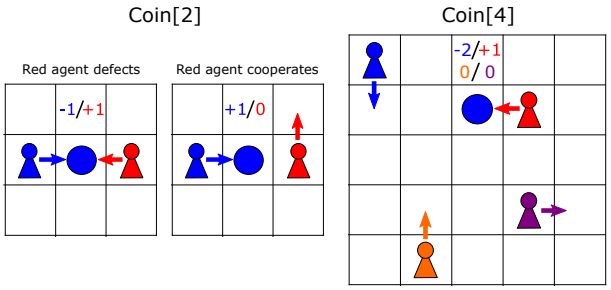


Figure 10: *Coin[2]* and *Coin[4]* as used in the paper.

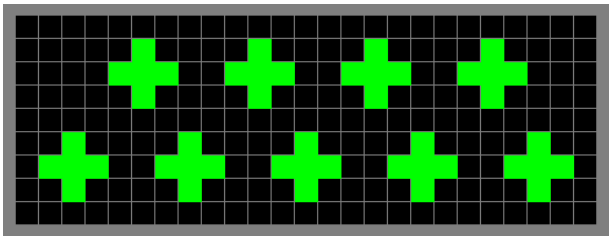


Figure 11: Domain layout with initial apple configuration used for *Harvest[6]* and *Harvest[12].*

A APPENDIX

A.1 Technical Details and Hyperparameters

A.1.1 Computing infrastructure. All training and test runs were performed in parallel on a computing cluster of 15 x86_64 GNU/Linux (Ubuntu 18.04.5 LTS) machines with i7-8700 @ 3.2GHz CPU (8 cores) and 64 GB RAM, where large-scale experiments with *Harvest[12]* took about 70 hours per run. We did not use any GPU in our experiments.

A.1.2 Hyperparameters. All common hyperparameters used by all MARL approaches in the experiments as reported in Section 6 in the paper are listed in Table 2. The final values were chosen based on a coarse grid search which finds a tradeoff between performance and computation w.r.t. *LIO* and *Naive Learning* in *Coin[2]* and *Harvest[6]*. We directly adopted the final values in Table 2 for all other approaches and domains from Section 5 and 6.

Similarly to $x_{token} = 1$ (Table 1), we set the gift reward of both *Gifting* variants (Section 5.2) to 1 as originally proposed in [22].

For *LIO*, we set the cost weight for learning the incentive function to 0.001 and the maximum incentive value R_{max} to the highest absolute penalty per domain (3 in *IPD*, 2 in *Coin[N]*, and 0.25 in *Harvest[N]*) as originally proposed in [42].

A.1.3 Neural Network Architectures. We coarsely tuned the neural network architectures from Section 5.3 in the paper w.r.t. performance and computation by varying the number of units per hidden layer {32, 64, 128} for $\hat{\pi}_i$ and $\hat{V}_i$. The number of hidden layers was varied between 1, 2, and 3, but significantly deeper architectures lead to detoriating performance (possibly due to vanishing gradients). All *MATE* variants, *Naive Learning*, and both *Gifting* variants only use $\hat{\pi}_i$ and $\hat{V}_i$ as neural networks. The policies $\hat{\pi}_i$ of both *Gifting* variants have an additional output unit for the gifting action, which is also part of the softmax activation.

The incentive function network of *LIO* has the same hidden layer architecture as $\hat{\pi}_i$ and $\hat{V}_i$. In addition, the joint action of the $N-1$ other agents is concatenated to the flattened observations before being input into the incentive function which outputs an $N-1$ dimensional vector. The output vector is passed through a sigmoid function and multiplied with R_{max} (Section A.1.2) afterwards.

Using ELU or ReLU activation did not make any significant difference for any neural network, thus we stick to ELU throughout the experiments.

A.2 Domain Details

A.3 IPD

An *IPD* episode consisted of 150 iterations similar to [10]. The gifting action of *Gifting* is treated as randomly picking C or D (simply picking C for gifting had the same effect though).

As a fully observable domain with just one opponent, all PI approaches use global communication, where each agent exchanges messages with the other respective agent.

A.4 Coin[N]

We adopted the setup of [10] in *Coin[2]* as shown in Fig. 10 with the same rules and reward functions. The order of executed actions is randomized such that situations, where two agents simultaneously step on a coin, lead to to an expected payoff of +1 for the red agent and -1 for the blue agent (Fig. 10 left). In addition, we extended the domain to 4 agents in *Coin[4]* (Fig. 10 right).

Since all agents are able to see each other's positions (although not being able to distinguish agents by color) all PI approaches use global communication, where each agent exchanges messages with $N-1$ other agents.

All agents are able to move freely and grid cell positions can be occupied by multiple agents. Any attempt to move out of bounds is treated as "do nothing" action.

A.5 Harvest[N]

We adopted the setup of [26] in *Harvest[6]* and *Harvest[12]* as shown in Fig. 11 with the same dynamics and apple regrowth rates. The initial apple configuration in Fig. 11 was used for both *Harvest[6]* and *Harvest[12]* to evaluate all MARL approaches in the absence and presence of social pressure respectively.

In addition to the original reward function, we added a time penalty of 0.01 for each agent at every time step t to increase pressure. Our agents are able to observe the environment around their 7×7 area and have no specific orientation. Thus, they have 4 separate actions to tag all neighbor agents which are either north, south, west, or east of them.

While *LIO* uses global all-to-all communication in *Harvest[N]*, all *MATE* and *Gifting* variants use local communication, where agents can only communicate with neighbor agents that are in their respective 7×7 field of view.

All agents are able to move freely and grid cell positions can be occupied by multiple agents. Any attempt to move out of bounds is treated as "do nothing" action.

Table 2: Common hyperparameters and their respective final values used by all algorithms evaluated in the paper. We also list the numbers that have been tried during development of the paper.

Hyperparam.	Final Value	Numbers/Range	Description
K	10	{1, 5, 10, 20}	Number of episodes per epoch.
E	5000	{2000, 5000, 10000}	Number of epochs. E was gradually increased to assess the stability of the learning progress until convergence.
α	0.001	{0.001}	Learning rate. We used the default value of ADAM in `torch` without further tuning.
Clip norm	1	{1,∞}	Gradient clipping parameter. Using a clip norm of 1 leads to better performance than disabling it with ∞.
λ	1	{0, 1}	Trace parameter for TD(λ) learning of the critic.
γ	0.95 (*IPD*, *Coin[N]*) 0.99 (*Harvest[N]*)	{0.9, 0.95, 0.99}	Discount factor for the return $G_{t,i}$. Any value ≥ 0.95 would have been sufficient.
$\lvert\tau_{t,i}\rvert$	1	{1, 5, 10}	Local history length. It was set to 1 to reduce computation because the other values did not significantly improve performance.