Autonomous Agents and Multi-Agent Systems (2024) 38:34
https://doi.org/10.1007/s10458-024-09666-5

Emergent cooperation from mutual acknowledgment exchange in multi-agent reinforcement learning

Thomy Phan[1,2] · Felix Sommer[2] · Fabian Ritz[2] · Philipp Altmann[2] · Jonas Nüßlein[2] · Michael Kölle[2] · Lenz Belzner[3] · Claudia Linnhoff-Popien[2]

Accepted: 2 July 2024 / Published online: 11 July 2024

Abstract
Peer incentivization (PI) is a recent approach where all agents learn to reward or 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 incorporated 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 exchange acknowledgment tokens as incentives to shape individual rewards mutually. All agents condition their token transmissions on the locally estimated quality of their own situations based on environmental rewards and received tokens. 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 deviate from the protocol and communication failures can occur. We also evaluate the sensitivity of MATE w.r.t. the choice of token values.

Keywords Multi-agent learning · Reinforcement learning · Mutual acknowledgments · Peer incentivization · Emergent cooperation

✉ Thomy Phan
thomy.phan@usc.edu

[1] University of Southern California, Los Angeles, USA

[2] LMU Munich, Munich, Germany

[3] Technische Hochschule Ingolstadt, Ingolstadt, Germany

1 Introduction

Many potential AI scenarios like autonomous driving [53], smart grids [14], or general IoT scenarios [11], where multiple autonomous systems coexist within a shared environment, can be naturally modeled as self-interested *multi-agent systems (MAS)* [7, 33]. 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 [16]. Conflict and competition are common in such systems due to opposing goals or shared resources [33, 41].

In order to maximize social welfare or efficiency in self-interested 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 a *social dilemma (SD)* [46]. SDs can be temporally extended to *sequential social dilemmas (SSD)* to model more realistic scenarios [30].

Multi-agent reinforcement learning (MARL) has become popular for modeling individually rational agents in SDs and SSDs to examine emergent behavior [7, 19, 30, 41, 48]. 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 toward other agents, which leads to defective behavior [3, 16, 30, 63].

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 [7, 22, 29, 60]. This can cause agents to defect from mutual cooperation, as studied extensively for the Prisoner's Dilemma [3, 16, 30, 46]. To mitigate this problem, some approaches propose to adapt the learning rate based on the outcome [6, 37, 66] or to incorporate information on other agents' adaptations, like gradients or opponent models [16, 27, 32]. These approaches are either tabular or require *full observability* to perceive each other's behavior and thus do not scale to complex domains. Furthermore, some approaches require knowledge about other agents' objectives to estimate their degree of adaptation therefore violating privacy [16, 32].

Another reason for mutual defection is the *reward structure*, which was found to be crucial for social intelligence [30, 54]. Prior work has shown that adequate reward formulations can lead to emergent cooperation in particular domains [4, 12, 13, 24, 42]. 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 [24, 26, 27, 68]. *Peer incentivization (PI)* is a distributed approach where all agents learn to reward or penalize each other, which often leads to emergent cooperation [36, 51, 64, 68]. Current PI mechanisms implicitly assume a flawless communication channel in order to exchange rewards. These rewards are assumed to be simply incorporated into the learning process without any chance to respond with feedback. Furthermore, most PI approaches rely on *global information* like joint actions [68], a central market function [51], or publicly available information [64], which limits scalability and applicability to real-world scenarios where only local information is accessible.

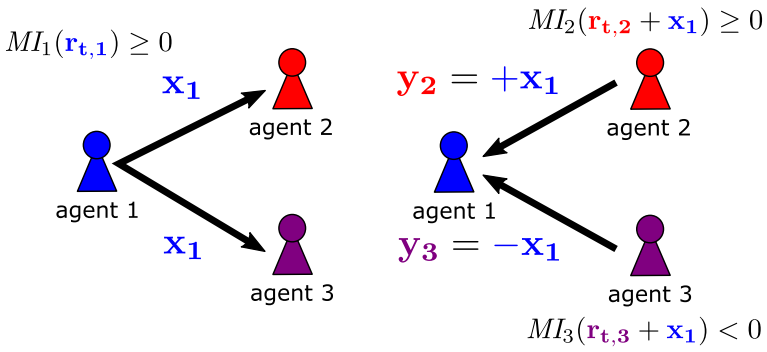
Once emergent cooperation has been achieved, it needs to be *maintained* to withstand social pressure, such as the *tragedy of the commons*, where many agents compete for scarce resources such that the outcome is less efficient than possible [30, 41] or disturbances like protocol defections or communication failures [3, 10]. Thus, *reciprocity* is important to establish *stable cooperation*, where social welfare is maintained over time without

deterioration by adequately responding to both cooperative and defective opponent behavior [2, 3, 47]. While reciprocity has already been considered in some prior learning rules [6, 16, 32, 34], there has been very little attention in most PI approaches, where agents are only able to exchange positive rewards to reach a consensus for cooperation—without any penalization mechanism against potential exploitation [36, 51, 68]. The lack of reciprocity at the reward level can, therefore, lead to naive cooperation in PI, which can be easily destabilized [28].

So far, penalization via negative rewards have been mostly provided by the environment rather than as a PI-based incentive [16, 28, 31]. However, the vast majority of SSD work studies specialized environments like Harvest or Cleanup that do not yield any negative reward for defective behavior, as defection only affects the temporal dynamics of the respective environment, such as being stunned or reducing the regrowth rate of resources [8, 18, 23–25, 27, 30, 36, 40, 41, 49, 51, 68]. While this indirectly affects the whole MAS, there is no explicit penalization of particular agents [24, 41]. Therefore, current PI research is mainly biased toward non-penalizing environments and approaches that lack reward-level reciprocity in general.

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 exchange acknowledgment tokens as incentives to shape individual rewards mutually. All agents condition their token transmissions on the locally estimated quality of their own situations based on environmental rewards and received tokens. 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 can locally estimate the long- or short-term quality of its own situation based on environmental rewards and received tokens.
- The MATE communication protocol and reward formulation using monotonic improvement estimation. The two phases of MATE ensure reward-level reciprocity, where



(a) Request phase (b) Response phase

Fig. 1 MATE protocol example. **a** If agent 1 estimates the monotonic improvement $MI_1(r_{t,1}) \geq 0$ of its own 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

agents get rewarded for accepted acknowledgment requests but penalized for rejected ones.
- 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 anomalously deviate from the protocol and communication failures can occur. We also evaluate the sensitivity of MATE w.r.t. the choice of token values.

This paper is an extended and revised version of our prior work [44], which was presented at the *21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*. The main extensions are more detailed discussions regarding practicability and reciprocity, additional experiments examining the sensitivity of MATE w.r.t. the choice of token values, and a discussion of limitations and prospects to address them.

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, $\langle r_{t,i} \rangle_{i \in \mathcal{D}} = \mathcal{R}(s_t, a_t) \in \mathbb{R}^N$ is the joint reward, $\mathcal{Z}$ is a set of local observations $z_{t,i}$ for each agent $i \in \mathcal{D}$, and $\Omega(s_t) = z_t = \langle z_{t,i} \rangle_{i \in \mathcal{D}} \in \mathcal{Z}^N$ is the joint observation of state s_t. Each agent i maintains a local *history* $\tau_{t,i} \in (\mathcal{Z} \times \mathcal{A}_i)^t \cdot$ $\pi_i(a_{t,i} | \tau_{t,i})$ 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, e.g., based on spatial, perceptional, or functional relationships, as suggested in [69]. A stochastic game M is *fully observable* when each agent $i \in \mathcal{D}$ is able to perceive the true state s_t and, thus, all other agents $j \neq i$ and their respective actions $a_{t,j}$ at every time step t. In such fully observable games, we assume $\mathcal{N}_{t,i} = \mathcal{D} - \{i\}$. However, the reverse statement does not hold, as $\mathcal{N}_{t,i} = \mathcal{D} - \{i\}$ does not necessarily imply that the game is fully observable, e.g., as in the Coin environment described in Sect. 5.1.2. Note that despite the reward function $\mathcal{R}$ depending on the true state s_t, each agent $i \in \mathcal{D}$ only perceives its corresponding output $r_{t,i}$ without explicit access or knowledge of $\mathcal{R}$. Furthermore, agents cannot uniquely deduce the full joint action from the obtained rewards in general.

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 with discount factor $\gamma \in [0, 1)$ and $\pi = \langle \pi_j \rangle_{j \in \mathcal{D}}$ is the *joint policy* of the MAS. In practice, the global state s_t is not directly observable for any agent i such that V_i^{π} is approximated with local information, i.e., $\tau_{t,i}$ instead [26, 30, 36, 41].

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 initial state s_0.

The goal of agent i is to find a *best response* π_i^* with $V_i^{\pi_i^*} = V_i^* = max_{\pi_i} V_i^{\langle \pi_i, \pi_{-i} \rangle}$ for all $s_t \in \mathcal{S}$, where π_{-i} is the joint policy *without* agent i. A *Nash equilibrium* is a solution concept where all local policies are best responses π_i^* to each other such that no agent can improve its value by deviating from its policy [3, 47, 63]. In SDs and SSDs, Nash equilibria do not maximize the efficiency (U) of a MAS; therefore, individually rational agents generally fail to learn cooperative behavior [2, 3, 10, 16, 30].

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 Sect. 2.3 [16, 60, 69]. *Naive (independent) learning* induces non-stationarity due to simultaneously adapting agents, which continuously changes the environment dynamics [22, 29, 33]. Therefore, naive learning can lead to overly greedy and exploitative policies which defect from any cooperative behavior [16, 30].

2.3 Policy gradient reinforcement learning

Policy gradient RL is a popular approach to approximate best responses π_i^* for each agent i [16, 35, 68]. A function approximator $\hat{\pi}_{i,\theta_i} \approx \pi_i^*$ with parameter vector θ_i is trained using gradient ascent on an estimate of $J = \mathbb{E}_\pi[G_{0,i}]$ [67]. Most policy gradient methods use gradients g of the following form [59]:

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

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 [16]. For simplicity, we omit the parameter indices θ_i, ω_i and write $\hat{\pi}_i$, $\hat{V}_i$ instead.

3 Related work

3.1 Multi-agent reinforcement learning in social dilemmas

MARL is a long standing research field with rapid progress and success in challenging domains [7, 33, 60, 65]. Different studies have been conducted on various complex SSDs, where interesting phenomena like group hunting, attacking and dodging, or flocking have been observed [19, 20, 28, 30, 41, 48]. Independent MARL, like naive learning, has been widely used in most studies to model agents with individual rationality [16, 60].

3.2 Non-stationarity in multi-agent reinforcement learning

Non-stationarity is one reason why naively learning agents fail to cooperate in SDs [7, 22, 29, 33, 60]. To mitigate this issue, different learning rates can be used depending on

the outcome [6, 37, 66]. Another approach is to incorporate "opponent awareness" into the learning rule by using or approximating other agents' gradients [16, 32]. 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.

3.3 Peer-incentivization

PI approaches have been introduced recently to encourage cooperative behavior in a distributed fashion via additional rewards. Multi-agent *Gifting* has been proposed in [36], 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 [68]. Both Gifting and LIO are unidirectional PI approaches, where agents have neither the ability to respond nor to penalize each other.

3.4 Peer-incentivization with global information

A market-based PI approach was devised in [51, 52], 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 decide on a joint market action additionally. Furthermore, this approach does not enable penalization of agents. Another approach based on public sanctioning has been proposed in [64]. 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.

3.5 Reciprocity

Strategies based on reciprocity are able to establish *stable cooperation* in SDs, i.e., where social welfare is maintained over time without deterioration, known as the *tragedy of the commons* [41], by adequately responding to other agents' actions [2, 3, 10, 47]. *Tit-for-Tat (TFT)* is a well-known reciprocal strategy for repeated 2-player games, which cooperates in the first time step and then imitates the last action of the other agent [47]. TFT is able to achieve and maintain mutual cooperation in simple games like the Iterated Prisoner's Dilemma while being able to defend itself against exploitation based on the following characteristics [2, 3]:

- *Niceness* Never be the first to defect.
- *Retaliation* Respond with defection after the other agent defected.

- *Forgiveness* Resume cooperation after the other agent cooperated, regardless of any prior defection.
- *Clarity* Be clear and recognizable.

Direct reciprocity (DR) is an analogous approach to TFT in evolutionary settings [62]. Agents in a population can choose either to cooperate or defect based on previous interactions and the probability of future interactions. However, TFT and DR require full observability of other agents' actions and a clear notion of cooperation and defection, which can only be assumed for simple games [30, 41].

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 [36, 51, 68] and independent adaptation of $\hat{\pi}_i$ and $\hat{V}_i$ [16, 30, 41]. Note that in our decentralized setting, each agent only stores its own information in $e_{t,i}$ in general without considering other agents' actions, observations, or rewards (unless that information is explicitly part of the observation, e.g., as in the Prisoner's Dilemma described in Sect. 5.1.1). The neighborhoods $\mathcal{N}_{t,i}$ are not stored in the experience tuples $e_{t,i}$ because they are only used for *communication* and not for updating the policy or value function parameters.

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 $m \leftarrow 1, E$ **do**
  Sample s_0 and set $\tau_{0,i}$ for all agents $i \in \mathcal{D}$
  for time step $t \leftarrow 0, T-1$ **do**
    for agent $i \in \mathcal{D}$ **do** ▷ Decision making in parallel
      Observe neighborhood $\mathcal{N}_{t,i}$
      $a_{t,i} \sim \hat{\pi}_i(a_{t,i}|\tau_{t,i})$
    end for
    $a_t \leftarrow \langle a_{t,i} \rangle_{i \in \mathcal{D}}$
    Execute joint action a_t
    $\langle r_{t,i} \rangle_{i \in \mathcal{D}} \leftarrow \mathcal{R}(s_t, a_t)$
    $s_{t+1} \sim \mathcal{P}(s_{t+1}|s_t, a_t)$
    $\langle z_{t+1,i} \rangle_{i \in \mathcal{D}} \leftarrow \Omega(s_{t+1})$
    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 $e_{t,i}$
    end for
  end for
  for agent $i \in \mathcal{D}$ **do** ▷ Update in parallel
    Update $\hat{\pi}_i$ and $\hat{V}_i$ via RL using all $e_{t,i}$ of episode m
  end for
end for

4.1 Monotonic improvement

After obtaining their respective experience tuples $e_{t,i}$, all agents can estimate the quality of their own situations by using a *monotonic improvement* measure $MI_{e_{t,i},\hat{V}_i}$ 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 either be the original environmental reward $r_{t,i}$ or some shaped reward, agent i can assume a monotonic improvement of its own 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 a monotonic improvement, in particular, to encourage agents to maintain their cooperative behavior instead of falling back to defective strategies.

MI_i represents a heuristic quality measure to predict if an agent i can rely on its environment represented by other agents $j \in \mathcal{N}_{t,i}$ without losing performance. Since MI_i can be

measured online, agent i is able to reciprocate at any time step t by either encouraging other agents j to reinforce their behavior if $MI_i(\hat{r}_{t,i}) \geq 0$ 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* quality of agent i's situation, i.e., how $\hat{r}_{t,i}$ compares to all rewards obtained so far. In case of uninformative rewards, e.g., $\hat{r}_{t,i} = 0$, the reward-based measure MI_i^{rew} can lead to misleading assessments since the underlying states may contribute to sparse or delayed rewards that are not considered at this point yet.

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. some *arbitrary reward* $\hat{r}_{t,i}$ and estimates the expected *long-term* quality of agent i's situation, 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 [57, 58]. Note that even uninformative rewards, e.g., $\hat{r}_{t,i} = 0$, can lead to informative values $MI_i^{TD}(\hat{r}_{t,i}) \neq 0$, given an adequate value function approximation $\hat{V}_i$, which requires sufficient exploration by all agents.

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

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 evaluates 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 neighbor 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 may be viewed 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 is sufficient 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 $\hat{r}_{req} = max\{\langle x_j \rangle_{j \in \mathcal{N}_{t,i}}\} \in \{0, x_{token}\}$ is the aggregation of all received requests x_j and $\hat{r}_{res} = min\{\langle y_j \rangle_{j \in \mathcal{N}_{t,i}}\} \in \{-x_{token}, 0, x_{token}\}$ 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 summation over all requests or responses, respectively, we prefer *max* and *min* aggregation to prevent single neighbor agents from being "voted out" by all other agents in $\mathcal{N}_{t,i}$. For example, if only a single neighbor agent responded with a negative token, a linear summation would weigh the positive responses more than the single negative case, therefore accepting isolated cases of dissatisfaction, which can spread in later iterations and consequently destabilize overall cooperation [2, 3, 10]. Thus, our reward formulation can push the interaction towards stable cooperation and fairness 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 measure 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 22 in Algorithm 1.

Algorithm 2 Mutual acknowledgment token exchange (MATE).

procedure $MATE(MI_i, \hat{V}_i, \mathcal{N}_{t,i}, \tau_{t,i}, e_{t,i})$
  $\hat{r}_{req} \leftarrow 0$, $\hat{r}_{res} \leftarrow 0$
  if $MI_i(r_{t,i}) \geq 0$ **then**
    Send acknowledgment request $x_i = x_{token}$ to all $j \in \mathcal{N}_{t,i}$
  end if
  for neighbor agent $j \in \mathcal{N}_{t,i}$ **do** ▷ Respond to requests
    if request x_j received from j **then**
      $\hat{r}_{req} \leftarrow max\{\hat{r}_{req}, x_j\}$
      if $MI_i(r_{t,i} + x_j) \geq 0$ **then**
        Send response $y_i = +x_j$ to agent j
      else
        Send response $y_i = -x_j$ to agent j
      end if
    end if
  end for
  if $MI_i(r_{t,i}) \geq 0$ **then** ▷ If requests have been sent before
    $\hat{r}_{res} \leftarrow 1$
    for neighbor agent $j \in \mathcal{N}_{t,i}$ **do** ▷ Receive responses
      if response y_j received from j **then**
        $\hat{r}_{res} \leftarrow min\{\hat{r}_{res}, y_j\}$
      end if
    end for
  end if
  return $r_{t,i} + \hat{r}_{req} + \hat{r}_{res}$ ($\hat{r}_{t,i}^{MATE}$ as defined in Eq. 4.2)
end procedure

4.3 Conceptual discussion of MATE

4.3.1 Practicability

MATE aims to incentivize all agents to learn cooperative behavior with a decentralized two-phase communication protocol. Agents using MATE 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 $\mathcal{N}_{t,i}$ thus do not require knowledge about other agent's objectives, or central instances like market functions or public information, as suggested in [16, 32, 35, 51, 64]. Locality of information is more practicable in real-world scenarios as global communication is typically expensive or infeasible, and disturbances mainly occur locally and, therefore, should not affect the whole MAS [61]. As mentioned above, MATE does not directly reveal an agent's objective due to merely exchanging acknowledgment tokens x_{token} instead of actual environment rewards $r_{t,i}$, learned values $\hat{V}_i(\tau_{t,i})$, or TD residuals. This can be useful for open scenarios like ad-hoc teamwork or IoT settings, where arbitrary agents

can join the system without revealing any private information or depending on central instances [5, 56]. Since MATE only modifies the environment reward for independent learning, our approach does not depend on any particular RL or distributed optimization algorithm.

4.3.2 Reciprocity

In contrast to Gifting and LIO, MATE ensures reward-level reciprocity in order to achieve *and* maintain emergent cooperation. While behavioral adaptation through RL is generally slow [21], MATE is able to respond immediately using rewards or penalties. Therefore, MATE exhibits the characteristics listed in Sect. 3.5 given that all agents use $\hat{r}_{t,i}^{MATE}$ according to Eq. 5 for adaptation:

- *Niceness* The request phase of MATE only uses positive rewards $x_{token} > 0$ and thus never defects first at the reward level.
- *Retaliation* MATE enables penalization of other agents by explicitly rejecting acknowledgment requests when $MI_i(r_{t,i} + x_{token}) < 0$, which has an immediate 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.
- *Forgiveness* MATE does not keep track of previous penalizations therefore being able to respond positively to any request as long as $MI_i(r_{t,i} + x_{token}) \geq 0$.
- *Clarity* MATE, according to Fig. 1 and Algorithm 2, defines a simple and easily recognizable communication protocol.

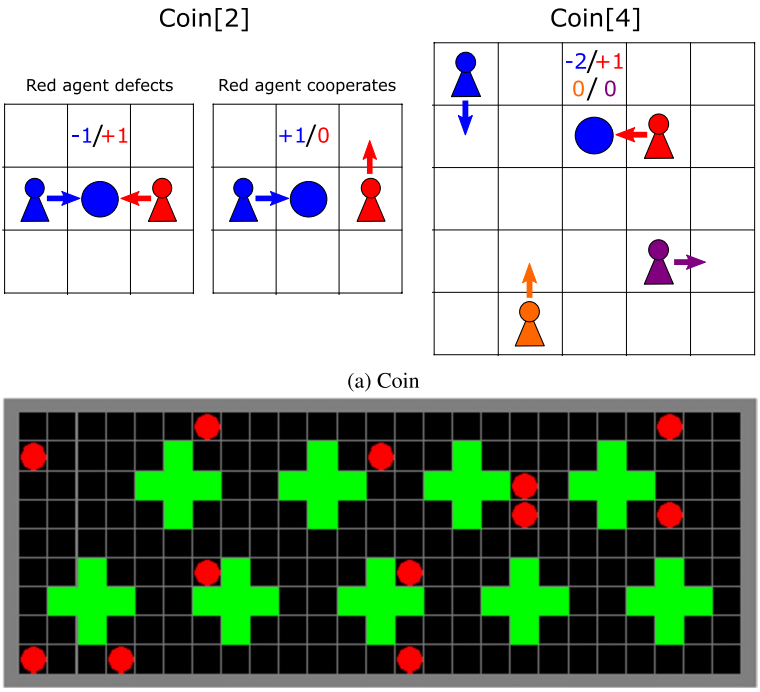
In contrast to TFT and DR, as described in Sect. 3.5, MATE is devised for general stochastic games; thus, neither assumes full observability of other agents' actions nor a clear notion of cooperation and defection, which is not trivial in complex domains [30, 41]. Instead, MATE uses MI_i to evaluate its local surroundings for adequate responses on the reward-level. Thus, MATE can be regarded as a reciprocal approach to self-interested MARL at a larger scale than TFT or DR.

4.3.3 Acknowledgment tokens

In this paper, we focus on fixed token values x_{token} to simplify evaluation and to focus on the main aspects of our approach, like [36]. The choice of x_{token} determines the degree of reciprocity by defining the reward and penalty scale. 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. In Sect. 6.4, we evaluate the sensitivity of MATE w.r.t. the choice of x_{token} in different domains. An adaptation of x_{token} to more flexible values, like in LIO [68], is left for future work. We note that agent-wise adaption of x_{token}, as discussed later in Sect. 7.3, might affect clarity according to Sect. 4.3.2, though.

4.3.4 Complexity

MATE scales with $\mathcal{O}(4(N-1))$ in the worst case according to Algorithm 2, if $\mathcal{N}_{t,i} = \mathcal{D} - \{i\}$ and $MI_i(r_{t,i}) \geq 0$ for all agents. In this particular setting, all agents would send $N-1$ requests, receive $N-1$ requests, respond positively to these requests, and receive $N-1$ positive responses. Other PI approaches like LIO or Gifting have a worst-case scaling of $\mathcal{O}(2(N-1))$ for sending and receiving rewards because they lack a response phase. Since MATE scales linearly w.r.t. N, it can still be considered feasible compared to alternative PI approaches, which scale exponentially [51]. Furthermore, the neighborhood size is typically $|\mathcal{N}_{t,i}| \ll N$ in practice such that the worst-case complexity becomes negligible in most cases.



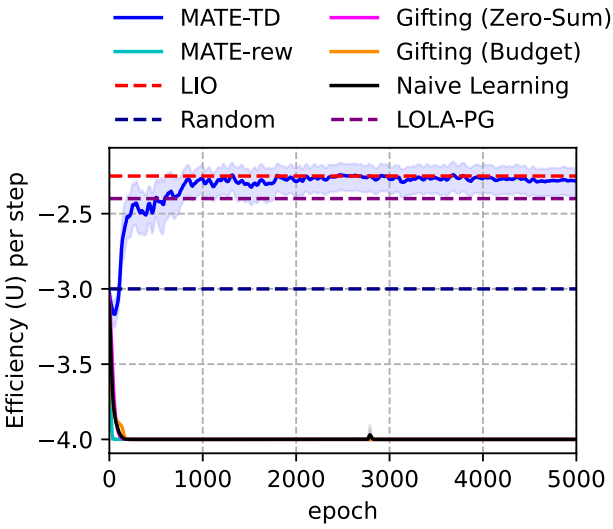
(a) Coin

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

Fig. 2 SSD environments for evaluation: **a** In *Coin[N]*, each agent gets a reward of +1 when collecting a coin. However, other agents are penalized with − 2 when the collected coin does not match with the collecting agent's color. **b** In *Harvest[N]*, 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 (Color figure online)

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

(a) Prisoner's Dilemma payoffs



(b) Efficiency in *IPD*

Fig. 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 taken from the respective papers [16, 68]

5 Experimental setup

5.1 Evaluation domains

We implemented three SD domains based on previous work [16, 36, 41]. 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. Further details are in Appendix A. Our code is available at https://github.com/thomyphan/emergent-cooperation.

5.1.1 Iterated prisoner's dilemma

The *Iterated Prisoner's Dilemma (IPD)* is a repeatedly played version of the 2-player Prisoner's Dilemma with the 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 the initial time step. The Nash equilibrium is to always defect (DD) with an average efficiency of $U = -2 - 2 = -4$ 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 $j \neq i$. The Prisoner's Dilemma is a stateless yet *fully observable* game since both agents are able to perceive each other's actions according to Sect. 2.1 and remember them throughout the IPD [2, 3, 10, 47, 62]. We use *IPD* for proof-of-concept to demonstrate that MATE can easily achieve mutual cooperation in a simple SD with a known Nash Equilibrium and a known global optimum.

5.1.2 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 [16, 31]. 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, which assesses the overall number of matching coin collections, 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, to assess if and how agents refrain from collecting other agents' coins. Despite $\mathcal{N}_{t,i} = \mathcal{D} - \{i\}$, our *Coin[N]* version is *partially observable* in general because agents cannot distinguish between other agents' colors. We use *Coin[N]* as an environment with global communication and negative rewards for particular agents, in contrast to non-penalizing environments like Cleanup, to assess stable cooperation and avoid bias in our evaluation, in contrast to [24, 36, 41, 68]. Note that the rewards depend on the color of each agent, according to Fig. 2a, b, and can differ depending on which agent collected a certain coin [16, 31, 44].

5.1.3 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 depends on the number of surrounding apples, where more neighbor apples lead to a higher regrowth rate [41]. 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 (U), we measure *equality (E)*, *sustainability (S)*, and *peace (P)* to analyze the degree of cooperation in more detail [41]:

$$E = 1 - \frac{\sum_{i\in\mathcal{D}} \sum_{j\in\mathcal{D}} |R_i - R_j|}{2N \sum_{i\in\mathcal{D}} R_i},$$

$$S = \frac{1}{N} \sum_{i\in\mathcal{D}} \Delta_i, \text{ where } \Delta_i = \mathbb{E}[t | r_{t,i} > 0],$$

$$P = N - \frac{1}{T} \sum_{i\in\mathcal{D}} \sum_{t=1}^{T} \mathbb{I}[\text{agent timed-out on time step } t]$$

Harvest[N] is a partially observable game because all agents only have a limited field of view to perceive and communicate with other agents. We use *Harvest[N]* to provide a large-scale environment with local communication to assess scalability and stable cooperation [24, 36, 41].

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 set $x_{token} = 1$ by default. 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* [16].

In addition, we implemented *LIO* [68], the zero-sum and replenishable budget version of *Gifting* [36], 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 [16] in *IPD* and *Coin[2]* for comparison.

5.3 Neural network architectures and hyperparameters

We implemented $\hat{\pi}_i$ and $\hat{V}_i$ for each agent i as a *multilayer perceptron (MLP)*. Since *Coin[N]* and *Harvest[N]* are gridworlds, states and observations are encoded as multi-channel images, as proposed in [17, 30]. The observations of *IPD* are the vector-encoded joint actions of the previous time step [16]. The multi-channel images of *Coin[N]* and *Harvest[N]* were flattened before being fed into the MLPs of $\hat{\pi}_i$ and $\hat{V}_i$. All MLPs have two hidden layers of 64 units with ELU activation. 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 Appendix B.

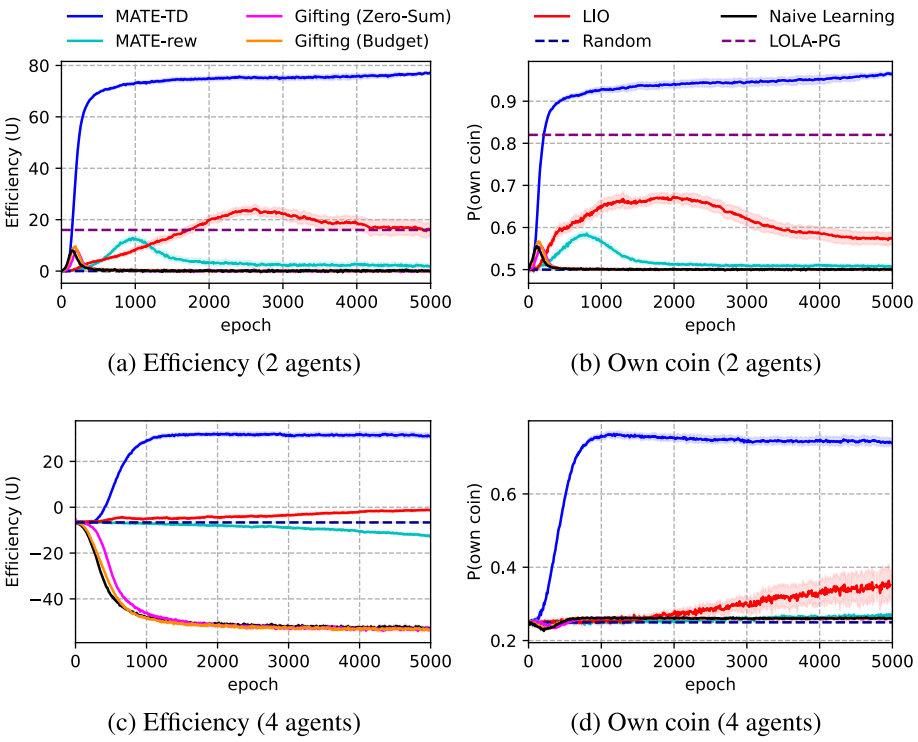


Fig. 4 Learning progress of MATE variants, LIO, Gifting variants, Naive Learning, and Random in *Coin[2]* and *Coin[4]*. The results of LOLA-PG are taken from the paper [16]

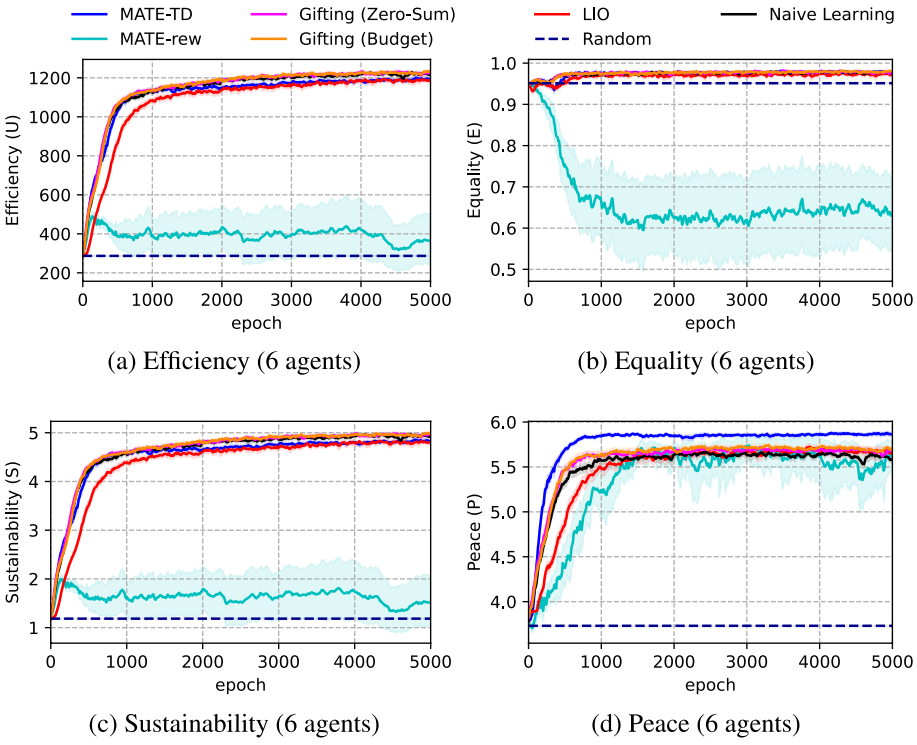
6 Results

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 1,000 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* 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 approach in *Coin[2]* (after *LOLA-PG* and *LIO*) and *Coin[4]*



(a) Efficiency (6 agents)

(b) Equality (6 agents)

(c) Sustainability (6 agents)

(d) Peace (6 agents)

Fig. 5 Learning progress of MATE variants, LIO, Gifting variants, Naive Learning, and Random in *Harvest[6]*

(after *Random*), but its "own coin" rate is similar to *Random*, meaning that one agents learns a more directed policy to collect more coins than the other but does not distinguish well between matching and non-matching coins due the short-sighted MI measure, according to Sect. 4.1. Both *Gifting* variants and *Naive Learning* perform similarly to *Random* in *Coin[2]*, where the chance of collecting one's matching coin is $\frac{1}{2}$, but are significantly less efficient than *Random* in *Coin[4]*, where each agent is more likely to be penalized due to any other agent collecting one's matching coin with a chance of $\frac{3}{4}$.

The results for *Harvest[6]* and *Harvest[12]* are shown in Figs. 5 and 6, respectively. 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 the second most peaceful after *MATE-rew*. Both *Gifting* variants are slightly more efficient, sustainable, and peaceful than *Naive Learning* in *Harvest[12]*, while *LIO* is progressing slower 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 significantly less equality than *Random*. *LIO*, both *Gifting* variants, and *Naive Learning* first improve w.r.t. all metrics but then exhibit a gradual decrease, indicating that agents become more aggressive and tag each other in order to harvest all apples alone, which is known as the *tragedy of the commons*

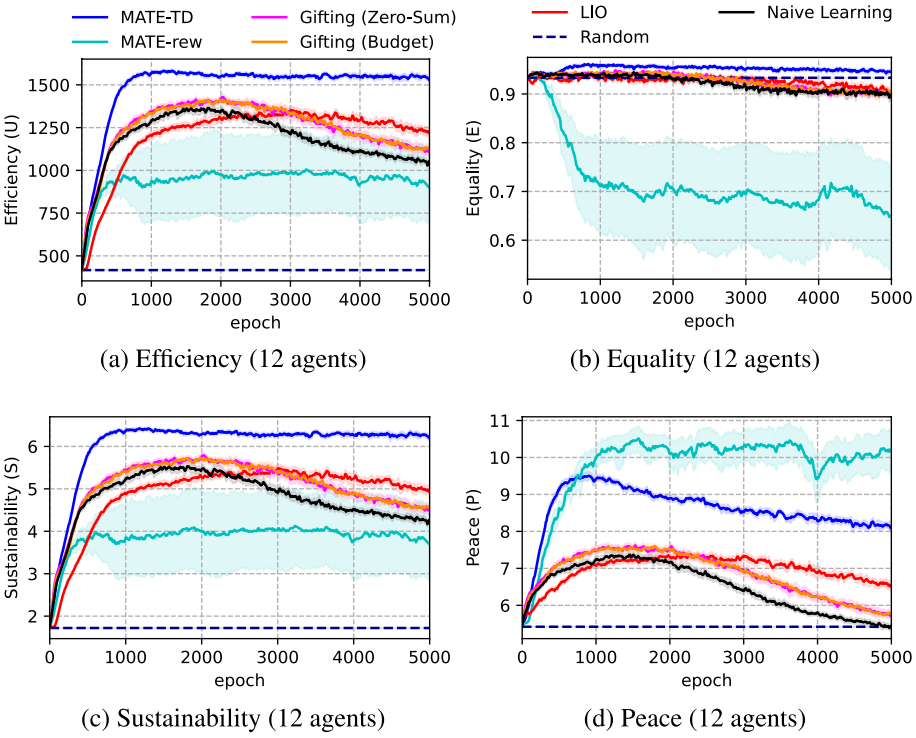


Fig. 6 Learning progress of MATE variants, LIO, Gifting variants, Naive Learning, and Random in *Harvest[12]*

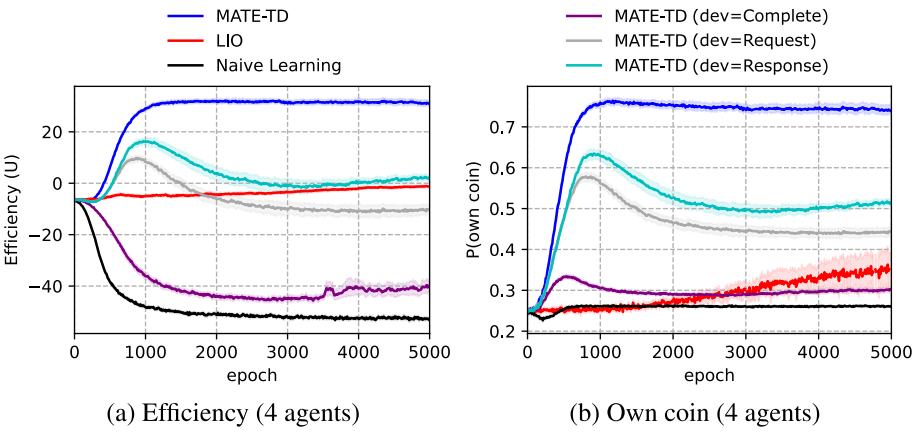


Fig. 7 Learning progress of MATE, anomalous MATE variants, LIO, and Naive Learning in *Coin[4]*

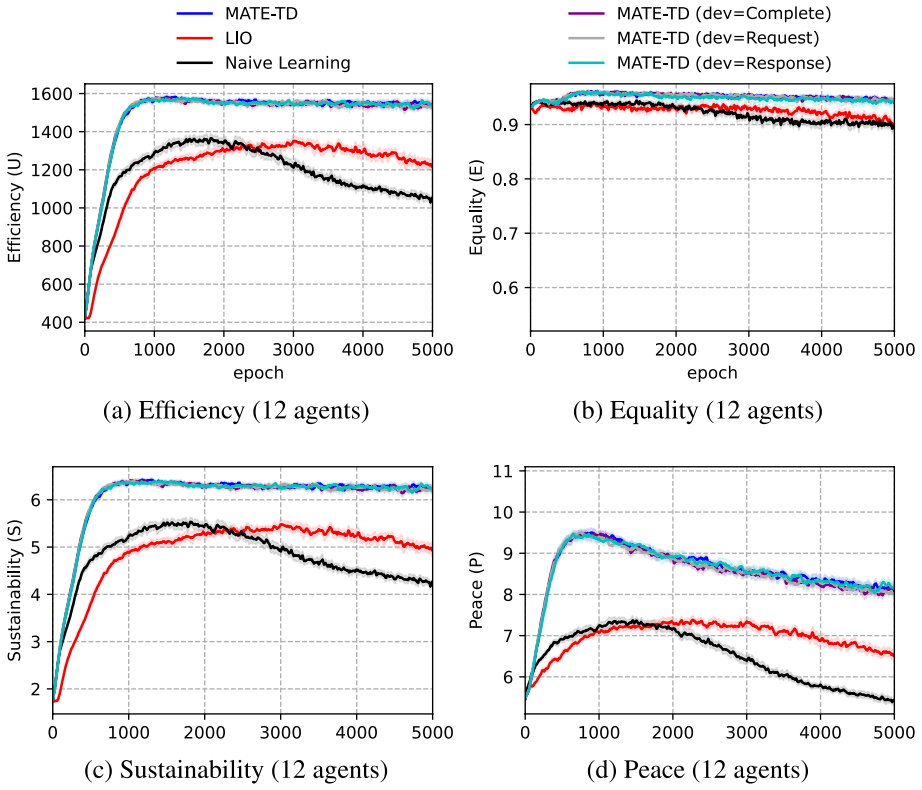


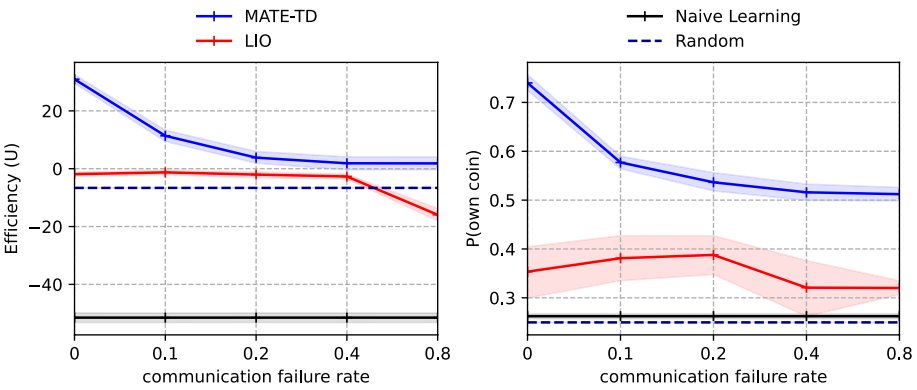
Fig. 8 Learning progress of MATE, anomalous MATE variants, LIO, and Naive Learning in *Harvest[12]*

[36, 41]. However, *MATE-TD* remains stable w.r.t. efficiency, equality, and sustainability in *Harvest[12]*, being able to maintain its high cooperation levels without any deterioration over time, indicating that *MATE-TD* is able to avoid the tragedy of the commons.

6.2 Robustness against anomalous protocol deviation

To evaluate the robustness of *MATE-TD* against anomalous protocol deviation, we introduce an anomalous agent $f \in \mathcal{D}$ which deviates from the communication protocol defined in Algorithm 2 and . 1 in one of the following ways:

- *Complete* The anomalous agent becomes a naive independent learner which does not participate in the communication rounds by skipping lines 16 and 17 in Algorithm 1. Thus, the anomalous agent f simply learns with its original reward $r_{t,f}$. This anomalous MATE variant lacks niceness, retaliation, and forgiveness according to Sect. 4.3.2.
- *Request* The anomalous agent f does not send any acknowledgment requests by skipping line 4 in Algorithm 2 and receives no responses in return. However, it can still



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

Fig. 9 Performance of MATE, LIO, Naive Learning, and Random in *Coin[4]* after 5,000 epochs w.r.t. different communication failure rates

receive requests from other agents $j \in \mathcal{N}_{t,f}$ and respond to them. Thus, the anomalous agent's reward is defined by $\hat{r}_{t,f}^{MATE} = r_{t,f} + \hat{r}_{req} = r_{t,f} + max\{\langle x_j \rangle_{j \in \mathcal{N}_{t,f}}\}$. This anomalous MATE variant lacks niceness according to Sect. 4.3.2.

- *Response* The anomalous agent f can send acknowledgment requests but ignores all responses by skipping lines 17–22 in Algorithm 2. In addition, it can receive requests from other agents $j \in \mathcal{N}_{t,f}$ and respond to them. Thus, the anomalous agent's reward $\hat{r}_{t,f}^{MATE}$ is the same as in the *Request* case above. This anomalous MATE variant does not lack any characteristics discussed in Sect. 4.3.2. However, the anomalous agent does not adapt its policy with the original MATE reward defined in Eq. 5.

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. In our experiments, we use the notation *MATE-TD (dev=X)* for the inclusion of an anomalous agent f using an anomalous MATE variant $X \in \{Complete, Request, Response\}$, deviating from the standard MATE protocol, as explained above.

The results for *Coin[4]* are shown in . 7. All anomalous *MATE-TD* variants are less efficient than *MATE-TD* but still more efficient with a higher "own coin" rate than *Naive Learning*. *MATE-TD (dev=Complete)* exhibits the least degree of cooperation. *MATE-TD (dev=Response)* is slightly more efficient than *LIO* and achieves a higher "own coin" rate. *MATE-TD (dev=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 . 8. All anomalous *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 probability or *communication failure rate* $\delta \in [0, 1)$, specifying that each agent can fail to send or receive a message with a chance of δ at every time step t. In particular, any of the following

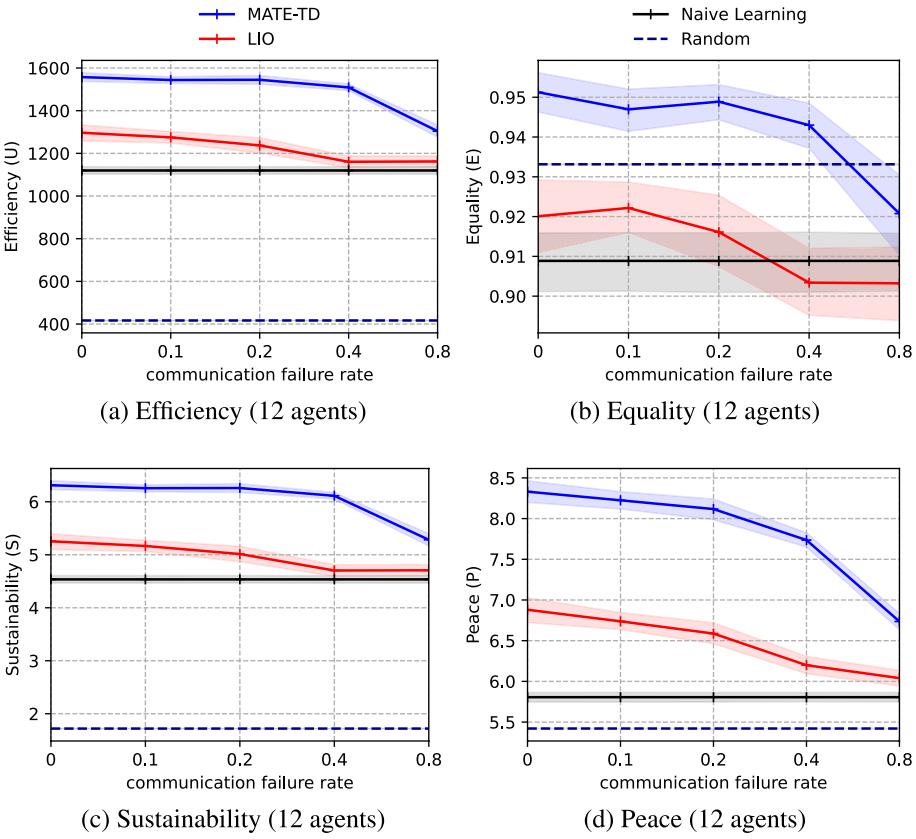


Fig. 10 Performance of MATE, LIO, Naive Learning, and Random in *Harvest[12]* after 5,000 epochs w.r.t. different communication failure rates

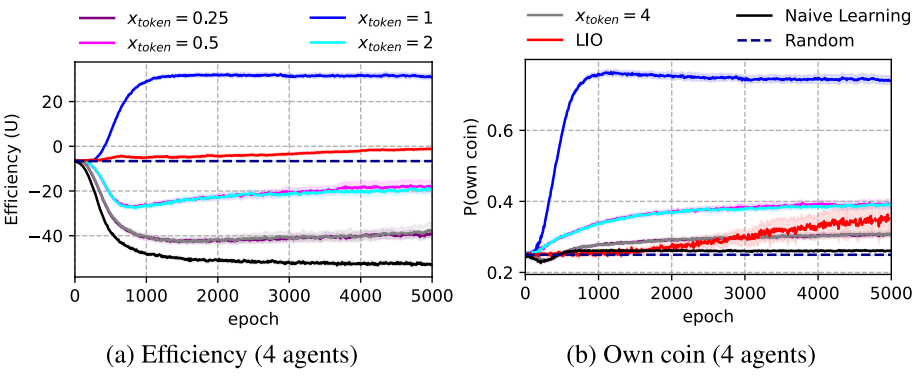
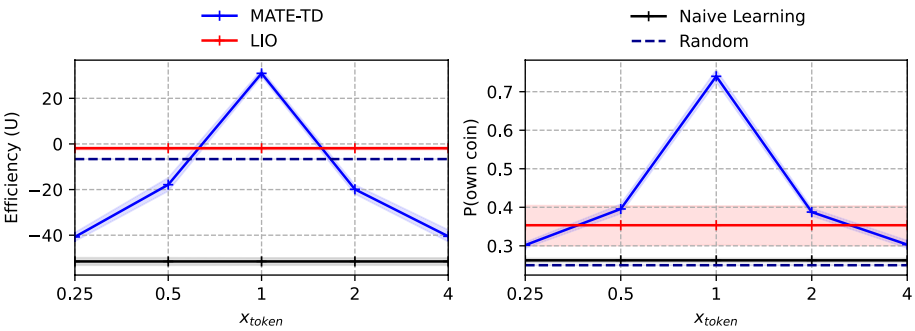
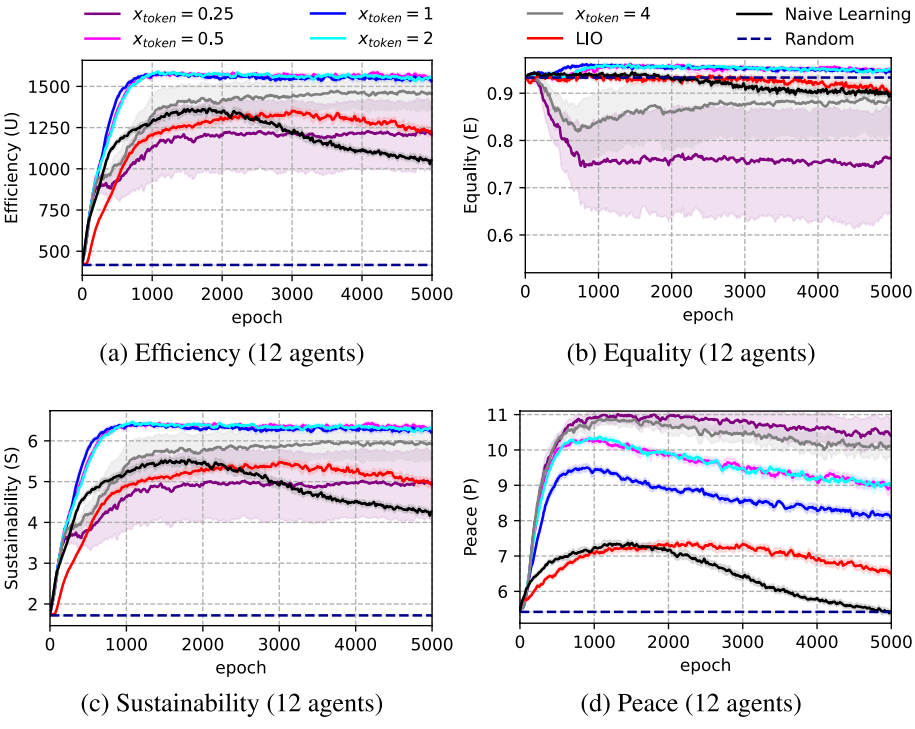


Fig. 11 Learning progress of MATE with $x_{token} \in \{0.25, 0.5, 1, 2, 4\}$, LIO, Naive Learning, and Random in *Coin[4]*



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

Fig. 12 Performance of MATE with $x_{token} \in \{0.25, 0.5, 1, 2, 4\}$, LIO, Naive Learning, and Random in *Coin[4]* after 5,000 epochs



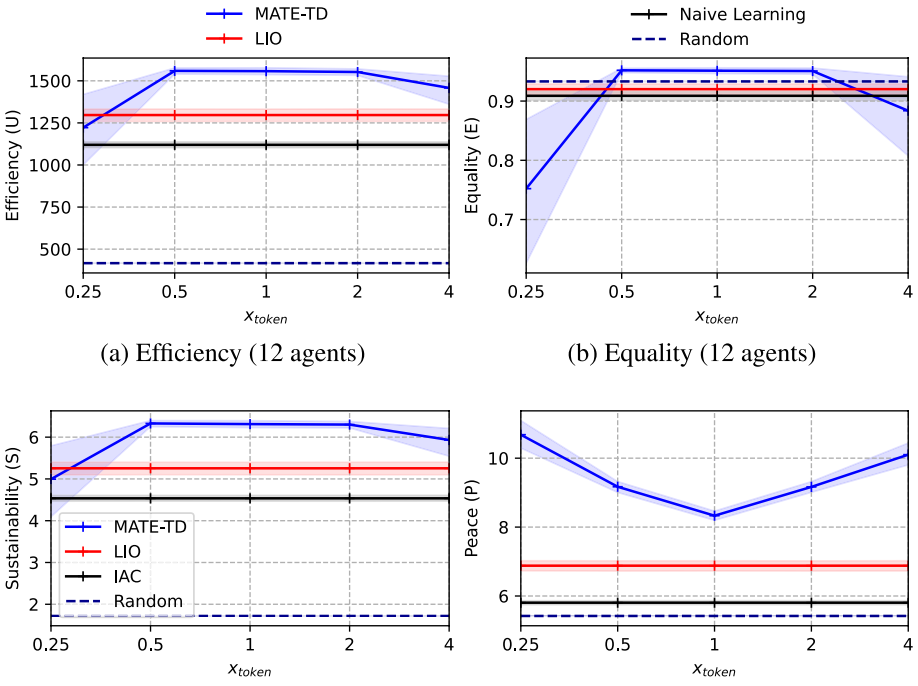
(a) Efficiency (12 agents) (b) Equality (12 agents)

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

Fig. 13 Learning progress of MATE with $x_{token} \in \{0.25, 0.5, 1, 2, 4\}$, LIO, Naive Learning, and Random in *Harvest[12]*

communication procedures from Algorithm 2 can be skipped with a probability of δ, where each message exchange between two agents can fail *independently* of all other exchanges:

- Sending an acknowledgement request, according to line 4.
- Receiving an acknowledgement request, according to lines 7–14.



(a) Efficiency (12 agents)

(b) Equality (12 agents)

(c) Sustainability (12 agents)

(d) Peace (12 agents)

Fig. 14 Performance of MATE with $x_{token} \in \{0.25, 0.5, 1, 2, 4\}$, LIO, Naive Learning, and Random in *Harvest[12]* after 5,000 epochs

- Sending an acknowledgement response, according to lines 9–13. Note that if a request is not received, then no response is sent. However if a request is successfully received, sending a response may still fail with a chance of δ.
- Receiving an acknowledgement response, according to lines 18–21.

We evaluate the final performance of *MATE-TD* and *LIO* at the end of training respectively 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 Sect. 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 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 [68]. In addition, we compare with *Naive Learning* and *Random* as non-communicating baselines.

The results for *Coin[4]* are shown in . 9. *MATE-TD* and *LIO* remain more efficient and cooperative than *Naive Learning* despite both approaches losing 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 "own coin" rates of *Naive Learning* and *Random*.

The results for *Harvest[12]* are shown in . 10. 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 $\delta \leq 0.4$.

6.4 Sensitivity to token values

To evaluate the sensitivity of *MATE-TD* w.r.t. the choice of x_{token}, we conduct experiments with $x_{token} \in \{0.25, 0.5, 1, 2, 4\}$. Setting $x_{token} = 0$ would reduce MATE to *Naive Learning*.

We report both the learning progress and the final performance at the end of training to assess stability and the relationship between x_{token} and the cooperation metrics explained in Sect. 5.1.

The results for *Coin[4]* are shown in Figs. 11 and 12. *MATE-TD* with $x_{token} = 1$ is the most efficient variant, achieving the highest "own coin" rate. *MATE-TD* is less efficient than *LIO* and *Random* when $x_{token} \neq 1$. However, *MATE-TD* with $x_{token} \in \{0.5, 2\}$ is able to achieve a higher "own coin" rate than *LIO* and *Random*. *MATE-TD* is always more efficient with a higher "own coin" rate than *Naive Learning*.

The results for *Harvest[12]* are shown in Figs. 13 and 14. All *MATE-TD* variants progress stably w.r.t. efficiency and sustainability without any deterioration over time. *MATE-TD* achieves the highest efficiency, equality, and sustainability with $x_{token} \in \{0.5, 1, 2\}$ and is always the most peaceful variant for any x_{token}. When $x_{token} = 0.25$, *MATE-TD* is less efficient and sustainable than *LIO*, while achieving less equality than *LIO*, *Naive Learning*, and *Random*. *MATE-TD* with $x_{token} = 4$ also achieves less equality than *LIO*, *Naive Learning*, and *Random* but is more efficient, sustainable, and peaceful. *MATE-TD* achieves the highest degree of peace when $x_{token} \in \{0.25, 4\}$ with notably high variance in all other metrics.

7 Discussion

7.1 Experimental results

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 establish stable cooperation in a completely decentralized way despite the increased social pressure compared to *Harvest[6]*, where all alternative PI approaches easily learn to cooperate.

Estimating the monotonic short-term quality via MI_i^{rew} (Eq. 3) can be beneficial compared to random acting and to some extent to naive learning in *Coin[2]* (Fig. 4). However, MI_i^{rew} cannot consider long-term effects, which is detrimental for sparse or delayed reward settings, where individual situations are assessed misleadingly and therefore lead to less cooperative behavior than possible. Considering the monotonic long-term quality 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 avoid the tragedy of

the commons by stably maintaining 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]* (Figs. 4 and 6), where the cooperation levels deteriorate over time.

MATE is not affected by anomalous MATE protocol variants in *Harvest[12]*, where agents only communicate locally, while the cooperation level significantly decreases in *Coin[4]*, where any deviation from the protocol can affect the whole MAS due to global communication (Figs. 7 and 8). The anomalous MATE variants in *Coin[4]* emphasize the importance of appropriate penalization mechanisms as proposed in our reward formulation in Eq. 5 for immediate retaliation according to Sect. 4.3.2 and [2, 3, 10]. Niceness through initiation of the MATE protocol according to Sect. 3.5 is also important as anomalous MATE variants using the strategy *Response* lead to superior cooperation in *Coin[4]* than variants using *Request*. Forgiveness is always implicitly assumed except for the anomalous MATE variant *Complete*, which leads to the least cooperative behavior in *Coin[4]*.

MATE shows some robustness against communication failures in Figs. 9 and 10, 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 $\delta \geq 10\%$. Unlike MATE, LIO already deteriorates with much lower communication failure rates in *Harvest[12]* due to its dependence on global communication.

x_{token} is a key hyperparameter of MATE since it defines the reward and penalty scale, which determines the degree of reciprocity in the system. As noted in Sect. 4.3.3, setting x_{token} to the highest positive reward yields the best results w.r.t. most metrics, as shown in Figs. 11, 12, 13 and 14, except for peace in *Harvest[12]*. MATE is very sensitive w.r.t. the choice of x_{token} in *Coin[4]*, where only $x_{token} = 1$ leads to the highest level of cooperation. The lower x_{token}, the more often agents tend to defect similarly to naive learning. On the other hand, if $x_{token} > 1$, then a single agent often manages to "bribe" all other agents to move elsewhere in order to collect the coin on its own. In *Harvest[12]*, MATE is more robust w.r.t. choice of x_{token}, as any $x_{token} \in \{0.5, 1, 2\}$ leads to higher levels of cooperation than alternative approaches. However, setting $x_{token} = 0.25$ leads to the least degree of cooperation w.r.t. efficiency, equality, and sustainability. As indicated by the sustainability metric in Fig. 14c, low values of x_{token} can lead to a greedy collection of apples, since agents cannot compensate each other for backing off. However, when $x_{token} > 2$, then most agents are not sufficiently incentivized to collect apples anymore since rewarding each other via MATE for "doing nothing" is more profitable if $\mathcal{N}_{t,i} \neq \emptyset$. The equality and sustainability results in Fig. 14b, c indicate that only agents with $\mathcal{N}_{t,i} = \emptyset$ tend to greedily collect apples since they cannot be rewarded by the MATE protocol. Therefore, the range of appropriate values for x_{token} also depends on each agent's neighborhood in addition to the scale of the highest positive reward.

7.2 Limitations

Budget Balance

Similar to many PI approaches [51, 64, 68], MATE is not budget-balanced, i.e., the rewards generated through PI are not subtracted from the incentivizing agents' reward,

which artificially increases the overall reward circulation in the MAS, thus fundamentally changing the game [2, 3, 10, 47]. However, in contrast to other PI approaches, where rewards are aggregated via summation [51, 68], MATE reduces the effect of reward imbalance via max/min aggregation of tokens, according to Eq. 5, which restricts the potential worst-case imbalance in the MAS to $2Nx_{token}$ at most, instead of N^2x_{token} (the factor 2 accounts for the two-phase protocol of MATE).

Reward Currency

In our setting, all agents share the same currency, e.g., when collecting a coin or apple in *Coin[N]* or *Harvest[N]*, respectively, which always yields a reward of +1 for the collecting agent. If agents had different currencies, i.e., valued certain events differently, then individual token values and a (decentralized) currency conversion mechanism would be needed [2, 3, 10, 47].

Synchronous Communication

Similar to most PI approaches [24, 36, 51, 52, 68], MATE assumes synchronous communication per time step, which is not perfectly realistic due to latencies based on communication distances, channels, and disturbances [55, 61]. Asynchronous communication could affect the learning progress and may require an additional memory for exchanged tokens in addition to the action-observation history $\tau_{t,i}$ to explicitly learn the temporal relationship between tokens, other agents' behavior, and environmental dynamics.

Neighborhood Definitions

So far, we assumed predefined neighborhoods based on the spatial perception ranges, which is a reasonable assumption in most spatio-temporal domains [44, 69], where sensors and communication ranges are limited. However, we did not study the impact of varying neighborhood sizes systematically, which could affect the efficiency and robustness of MATE in addition to the token value definition, as mentioned in Sect. 6.4. Furthermore, we assumed homogeneity, where all agents have the same perception and communication range. An interesting direction for future work would be the evaluation of different neighborhood definitions, based on individual perception ranges, noisy sensors, and functional relationships, i.e., where agents can only perceive certain types of other agents.

Predefined Token Values

As discussed in Sect. 4.3.3 and experimentally evaluated in Sect. 6.4, the choice of token value x_{token} is crucial for the ability of MATE to achieve stable cooperation. While a default token value of $x_{token} = 1$ has been empirically shown to work well for standard benchmark environments [36, 44, 51, 52], any change in the environment, neighborhood definition, or reward scale could render the default choice ineffective. In the following Sect. 7.3, we will discuss the challenges and prospects of adaptive token values, which could mitigate the issues of predefined token values.

7.3 Challenges and prospects on adaptive token values

In cases where the reward function is not known a priori, the token value x_{token} needs to be learned and adapted with online experience. In addition to learning an adequate value for x_{token}, all agents need to *synchronize on the same token value* to avoid "bribery" or inequality of rewards, e.g., where one agent can send larger token values and, therefore, have a stronger influence on other agents. This poses a particular challenge in our decentralized SSD setting since agents generally do not have access to global communication, as in [24, 51, 68], or centralized instances, as in [25, 52, 64].

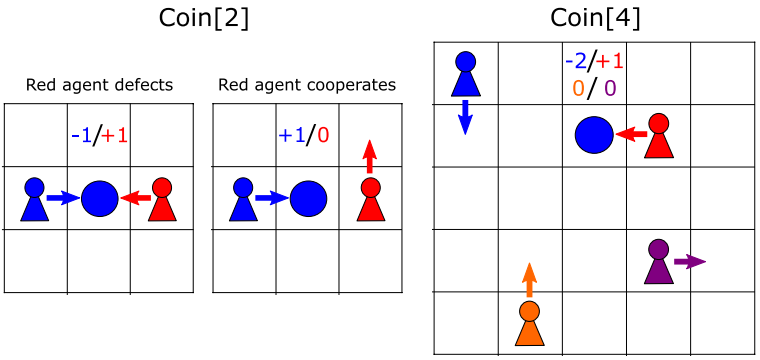


Fig. 15 *Coin[2]* and *Coin[4]* as used in the paper

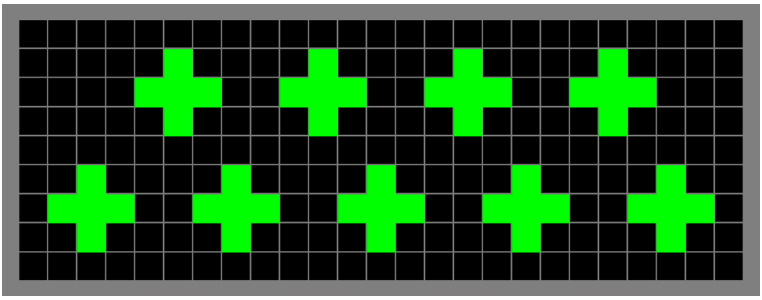


Fig. 16 Domain layout with initial apple configuration used for *Harvest[6]* and *Harvest[12]*

Another challenge is the potential change or drift in rewards, e.g., where the scale of rewards changes over time due to environmental or perceptional changes. Such changes require constant adaptation and synchronization.

A centralized way of learning and synchronizing token values can be implemented with a shared and periodically updated server to record the environmental rewards observed by all agents. To mitigate the necessity of constant accessibility for all agents, each agent can locally store its environmental reward to asynchronously update the central server and synchronize its individual token value, depending on periodic time slots, spatial distance to the server, or any locally detected change in rewards [25, 38, 64].

A decentralized way of learning and synchronizing token values could be the employment of consensus algorithms, where agents exchange their individually estimated mean rewards or token values to jointly agree on a common token value x_{token} [9]. There exist several consensus algorithms for estimating common values that are completely decentralized and only require local value estimation and communication [1, 39, 50, 55]. The consensus approach could be combined with LIO to learn individual token values per agent in order to accommodate different reward currencies for more general scenarios [44, 68].

8 Conclusion and future work

We presented MATE, a PI approach defined by a two-phase communication protocol to exchange acknowledgment tokens as incentives to shape individual rewards mutually. All agents condition their token transmissions on the locally estimated quality of their own situations based on environmental rewards and received tokens. MATE is completely decentralized and only requires local communication and information without knowledge about other agents' objectives or any public information. In addition to rewarding other agents, MATE enables penalization for reward-level reciprocity by explicitly rejecting acknowledgment requests, causing an immediate negative effect on the requesting agent's reward.

MATE was evaluated in the Iterated Prisoner's Dilemma, Coin, and Harvest. We compared the results to other PI approaches w.r.t. different cooperation metrics showing that MATE is able to achieve and maintain significantly higher levels of cooperation than previous PI approaches even in the presence of social pressure and disturbances like anomalous protocol variants or communication failures. While being rather sensitive w.r.t. the choice of token values, MATE always tends to learn more cooperative policies than naive learning thus being generally a more beneficial choice for self-interested MARL, when communication is possible to some extent at least.

MATE is suitable for more realistic scenarios, e.g., in ad-hoc teamwork or IoT settings with private information, where single agents can deviate from the protocol, e.g., due to malfunctioning or selfishness, and where communication is not perfectly reliable.

Future work includes the determination of appropriate bounds w.r.t. the choice of token values, the automatic adjustment of token values for more flexibility, e.g., by combining LIO and MATE, and an integration of emergent communication and consensus techniques to create more adaptive and intelligent agents with social capabilities [15, 54]. Furthermore, we want to explore the impact of neighborhood definitions and sizes to study the influence of certain agents on the overall cooperation as well as the reciprocal consequences, e.g., how a change in monotonic improvement by a single agent can cause neighborhood retaliation and to what extent [43, 45].

Appendix A Evaluation domain details

A.1 IPD

An *IPD* episode consists of 150 iterations similar to [16]. The gifting action of *Gifting* is treated as randomly picking C or D to avoid any bias (simply picking C for gifting has 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.2 Coin[N]

We adopt the setup of [16] in *Coin[2]* as shown in Fig. 15 with the same rules and reward functions. In addition, we extend the domain to 4 agents in *Coin[4]* (Fig. 15 right).

Since all agents are able to perceive each other's positions (albeit 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. The order of executed actions is randomized to resolve situations, where multiple agents step on a coin simultaneously.

A.3 Harvest[N]

We adopt the setup of [41] in *Harvest[6]* and *Harvest[12]* as shown in Fig. 16 with the same dynamics and apple regrowth rates. The initial apple configuration in Fig. 16 is used for both *Harvest[6]* and *Harvest[12]* to evaluate all MARL approaches in the absence and presence of social pressure respectively.

We modify the original reward function by adding a time penalty of 0.01 for each agent at every time step t to increase pressure. All agents are able to observe the environment around their 7×7 area and have no specific orientation. Thus, each agent has 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 all 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. The order of executed actions is randomized to resolve situations, where multiple agents attempt to collect an apple or tag each other simultaneously.

Appendix B Technical details

B.1 Hyperparameters

All common hyperparameters used by all MARL approaches in the experiments, as reported in Sect. 6, are listed in Table 1. The final values are chosen based on a coarse grid search to find a tradeoff between performance and computation for *LIO* and *Naive Learning* in *Coin[2]* and *Harvest[6]*. We directly adopt the final values in Table 1 for all other approaches and domains from Sect. 5 and 6.

Similarly to $x_{token} = 1$, we set the gift reward of both *Gifting* variants introduced in Sect. 5.2 to 1 as originally proposed in [36].

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 [68].

B.2 Neural network architectures

We coarsely tuned the neural network architectures from Sect. 5.3 w.r.t. performance and computation by varying the number of hidden layers {1, 2, 3} as well as the number of

Table 1 Common hyperparameters and their respective final values used by all algorithms evaluated in the paper. We also list the values and ranges that have been tried during development of the paper

Hyperparameter	Final Value	Values/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
# hidden layers	2	{1, 2, 3}	Number of hidden layers of the MLPs. See Sect. B.2
# units per hidden layer	64	{32, 64, 128}	Number of units per hidden layer. See Sect. B.2
Hidden layer activation	ELU	{ReLU, ELU}	Activation function used for the hidden layer outputs. See Sect. B.2
Optimizer	ADAM	{ADAM, RMSProp}	Gradient-based optimization algorithm for MLP training
α	0.001	{0.001}	Learning rate. We used the default value of ADAM in `torch` for all MLPs 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 $\hat{V}_i$
γ	0.95 (*IPD*, *Coin[NJ]*) 0.99 (*Harvest[NJ]*)	{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

units per hidden layer {32, 64, 128} for $\hat{\pi}_i$ and $\hat{V}_i$. All *MATE* variants, *Naive Learning*, and both *Gifting* variants use $\hat{\pi}_i$ and $\hat{V}_i$ as separate MLPs. 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} (Sect. B.1) afterwards.

Using ELU or ReLU activation does not make any significant difference for any MLP thus we stick to ELU throughout the experiments.

Author Contributions TP and FS designed and implemented the concepts. TP, FS, PA, JN, and LB discussed the concepts. TP, FS, FR, and LB provided and discussed related work. TP, FS, and FR designed and conducted the experiments. TP, FS, FR, MK, and CL discussed the results and visualized the data. All authors contributed to writing the manuscript.

Funding Open Access funding enabled and organized by Projekt DEAL. This work was partially funded by the Bavarian Ministry for Economic Affairs, Regional Development and Energy as part of a project to support the thematic development of the Institute for Cognitive Systems.

Availability of Data and Materials Our code is available at https://github.com/thomyphan/emergent-cooperation.

Declarations

Conflict of interest T.P. contributed to the work at LMU Munich and is now affiliated with the University of Southern California. F.S., F.R., P.A., J.N., M.K., and C.L. contributed to the work while affiliated with LMU Munich. L.B. contributed to the work while affiliated with Technische Hochschule Ingolstadt.

References

1. Amirkhani, A., & Barshooi, A. H. (2022). Consensus in multi-agent systems: A review. *Artificial Intelligence Review, 55*(5), 3897–3935.
2. Axelrod, R. (1984). *The Evolution Of Cooperation*. New York: Basic Books.
3. Axelrod, R., & Hamilton, W. D. (1981). The evolution of cooperation. *Science, 211*(4489), 1390–1396.
4. Babes, M., Munoz de Cote, E. & Littman, M. L. (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*, pp. 1389–1392. International Foundation for Autonomous Agents and Multiagent Systems.
5. Barrett, S., Stone, P., & Kraus, S. (2011). Empirical evaluation of Ad Hoc teamwork in the pursuit domain. In *The 10th international conference on autonomous agents and multiagent systems - volume 2*, AAMAS '11, pp. 567–574. International Foundation for Autonomous Agents and Multiagent Systems.

6. Bowling, M., & Veloso, M. (2002). Multiagent learning using a variable learning rate. *Artificial Intelligence, 136*(2), 215–250.
7. Buşoniu, L., Babuška, R., & De Schutter, B. (2008). Multi-agent reinforcement learning: An overview. *IEEE Transactions on Systems, Man, and Cybernetics-Part C: Applications and Reviews, 38*(2), 156–172.
8. Christoffersen, P. J., Haupt, A. A., & Hadfield-Menell, D. (2023). Get it in writing: Formal contracts mitigate social dilemmas in multi-agent RL. In *22nd international conference on autonomous agents and multiagent systems (AAMAS)*, AAMAS '23, pp. 448–456. International Foundation for Autonomous Agents and Multiagent Systems.
9. Conradt, L., & Roper, T. J. (2005). Consensus decision making in animals. *Trends in Ecology & Evolution, 20*(8), 449–456.
10. Dawkins, R. (2016). *The selfish gene: 40th (Anniversary)*. Oxford Landmark ScienceUK: OUP Oxford.
11. Deng, S., Xiang, Z., Zhao, P., Taheri, J., Gao, H., Yin, J., & Zomaya, A. Y. (2020). Dynamical resource allocation in edge for trustable internet-of-things systems: A reinforcement learning method. *IEEE Transactions on Industrial Informatics, 16*(9), 6103–6113.
12. Devlin, S., & Kudenko, D. (2011). Theoretical considerations of potential-based reward shaping for multi-agent systems. In *The 10th international conference on autonomous agents and multiagent systems*, pp. 225–232. ACM, International Foundation for Autonomous Agents and Multiagent Systems.
13. Devlin, S., Yliniemi, L., Kudenko, D., & Tumer, K. (2014). Potential-based difference rewards for multiagent reinforcement learning. In *Proceedings of the 2014 international conference on autonomous agents and multi-agent systems*, pp. 165–172. International Foundation for Autonomous Agents and Multiagent Systems.
14. Dimeas, A. L., & Hatziargyriou, N. D. (2010). Multi-agent reinforcement learning for microgrids. In *IEEE PES General Meeting*, pp. 1–8. IEEE.
15. Foerster, J., Assael, I. A., De Freitas, N., & Whiteson, S. (2016). Learning to communicate with deep multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems*, pp. 2137–2145. Red Hook: Curran Associates Inc.
16. Foerster, J. N., Chen, R. Y., Al-Shedivat, M., Whiteson, S., Abbeel, P., & Mordatch, I. (2018). Learning with opponent-learning awareness. In *Proceedings of the 17th international conference on autonomous agents and multiagent systems*, pp. 122–130. International Foundation for Autonomous Agents and Multiagent Systems.
17. Gupta, J. K., Egorov, M., & Kochenderfer, M. (2017). Cooperative multi-agent control using deep reinforcement learning. *Autonomous Agents and Multiagent Systems, 10642*, 66–83.
18. Guresti, B., Vanlioglu, A., & Ure, N. K. (2023). IQ-flow: Mechanism design for inducing cooperative behavior to self-interested agents in sequential social dilemmas. In *22nd international conference on autonomous agents and multiagent systems (AAMAS)*, AAMAS '23, pp. 2143–2151. International Foundation for Autonomous Agents and Multiagent Systems.
19. Hahn, C., Phan, T., Gabor, T., Belzner, L., & Linnhoff-Popien, C. (2019). Emergent escape-based flocking behavior using multi-agent reinforcement learning. volume ALIFE 2019: The 2019 Conference on Artificial Life of *ALIFE 2019: The 2019 Conference on Artificial Life*, pp. 598–605. MIT Press.
20. Hahn, C., Ritz, F., Wikidal, P., Phan, T., Gabor, T., & Linnhoff-Popien, C. (2020). Foraging swarms using multi-agent reinforcement learning. volume ALIFE 2020: The 2020 Conference on Artificial Life of *ALIFE 2020: The 2020 Conference on Artificial Life*, pp. 333–340. MIT Press.
21. Hennes, D., Morrill, D., Omidshafiei, S., Munos, R., Perolat, J., Lanctot, M., Gruslys, A., Lespiau, J.B., Parmas, P., Duéñez-Guzmán, E., & Tuyls, K. (2020). Neural replicator dynamics: Multiagent learning via hedging policy gradients. In *Proceedings of the 19th international conference on autonomous agents and multiagent systems*, AAMAS '20, pp. 492–501. International Foundation for Autonomous Agents and Multiagent Systems.
22. Hernandez-Leal, P., Kaisers, M., Baarslag, T., & De Cote, E. M. (2017). A survey of learning in multiagent environments: Dealing with non-stationarity. arXiv preprint arXiv:1707.09183.
23. Hua, Y., Gao, S., Li, W., Jin, B., Wang, X., & Zha, H. (2023). Learning optimal "Pigovian Tax" in sequential social dilemmas. In *22nd international conference on autonomous agents and multiagent systems (AAMAS)*, AAMAS '23, pp. 2784–2786. International Foundation for Autonomous Agents and Multiagent Systems.
24. Hughes, E., Leibo, J. Z., Phillips, M., Tuyls, K., Dueñez-Guzman, E., García Castañeda, A., Dunning, I., Zhu, T., McKee, K., Koster, R., et al. (2018). Inequity aversion improves cooperation in intertemporal social dilemmas. In *Proceedings of the 32nd international conference on neural information processing systems*, pp. 3330–3340. Red Hook: Curran Associates Inc.

25. Ivanov, D., Zisman, I., & Chernyshev, K. (2023). Mediated multi-agent reinforcement learning. In *Proceedings of the 2023 international conference on autonomous agents and multi-agent systems*, pp. 49–57. International Foundation for Autonomous Agents and Multiagent Systems.
26. Jaderberg, M., Czarnecki, W. M., Dunning, I., Marris, L., Lever, G., Castaneda, A. G., Beattie, C., Rabinowitz, N. C., Morcos, A. S., Ruderman, A., et al. (2019). Human-level performance in 3D multiplayer games with population-based reinforcement learning. *Science, 364*(6443), 859–865.
27. Jaques, N., Lazaridou, A., Hughes, E., Gulcehre, C., Ortega, P., Strouse, D. J., Leibo, J. Z., & De Freitas, N. (2019). Social influence as intrinsic motivation for multi-agent deep reinforcement learning. In: Kamalika, C., & Ruslan, S. (eds.) *Proceedings of the 36th international conference on machine learning*, volume 97 of *Proceedings of machine learning research*, pp. 3040–3049. PMLR, 09–15.
28. Köster, R., Hadfield-Menell, D., Hadfield, G. K., & Leibo, J. Z. (2020). Silly rules improve the capacity of agents to learn stable enforcement and compliance behaviors. In *Proceedings of the 19th international conference on autonomous agents and multiagent systems*, AAMAS '20, pp. 1887–1888. International Foundation for Autonomous Agents and Multiagent Systems.
29. Laurent, G. J., Matignon, L., Fort-Piat, L., et al. (2011). The world of independent learners is not Markovian. *International Journal of Knowledge-Based and Intelligent Engineering Systems, 15*(1), 55–64.
30. Leibo, J. Z., Zambaldi, V., Lanctot, M., Marecki, J., & Graepel, T. (2017). Multi-agent reinforcement learning in sequential social dilemmas. In *Proceedings of the 16th conference on autonomous agents and multiagent systems*, AAMAS '17, pp. 464–473. International Foundation for Autonomous Agents and Multiagent Systems.
31. Lerer, A., & Peysakhovich, A. (2017). Maintaining cooperation in complex social dilemmas using deep reinforcement learning. arXiv preprint arXiv:1707.01068.
32. Letcher, A., Foerster, J., Balduzzi, D., Rocktäschel, T., & Whiteson, S. (2019). Stable opponent shaping in differentiable games. In *International conference on learning representations*.
33. Littman, M. L. (1994). Markov games as a framework for multi-agent reinforcement learning. In *Machine learning proceedings 1994*, pp. 157–163. Morgan Kaufmann, San Francisco.
34. Littman, M. L. (2001). Friend-or-foe Q-learning in general-sum games. In *Proceedings of the eighteenth international conference on machine learning*, ICML '01, pp. 322–328. San Francisco: Morgan Kaufmann Publishers Inc.
35. Lowe, R., Wu, Y. I., Tamar, A., Harb, J., Pieter Abbeel, O., & Mordatch, I. (2017). Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in Neural Information Processing Systems*, vol. 30. Curran Associates, Inc.
36. Lupu, A., & Precup, D. (2020). Gifting in multi-agent reinforcement learning. In *Proceedings of the 19th international conference on autonomous agents and multiagent systems*, pp. 789–797. International Foundation for Autonomous Agents and Multiagent Systems.
37. Matignon, L., Laurent, G. J., & Le Fort-Piat, N. (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*, pp. 64–69. IEEE.
38. Müller, R., Illium, S., Phan, T., Haider, T., & Linnhoff-Popien, C. (2022). Towards anomaly detection in reinforcement learning. In *21st international conference on autonomous agents and multiagent systems (AAMAS)*, AAMAS '22, pp. 1799–1803. International Foundation for Autonomous Agents and Multiagent Systems.
39. Olfati-Saber, R., & Shamma, J. S. (2005). Consensus filters for sensor networks and distributed sensor fusion. In *Proceedings of the 44th IEEE conference on decision and control*, pp. 6698–6703. IEEE.
40. Orzan, N., Acar, E., Grossi, D., & Rădulescu, R. (2024). Emergent cooperation under uncertain incentive alignment. In *23rd international conference on autonomous agents and multiagent systems (AAMAS)*, AAMAS '24, pp. 1521–1530. International Foundation for Autonomous Agents and Multiagent Systems.
41. Perolat, J., Leibo, J. Z., Zambaldi, V., Beattie, C., Tuyls, K., & Graepel, T. (2017). A multi-agent reinforcement learning model of common-pool resource appropriation. In: *Proceedings of the 31st international conference on neural information processing systems*, NIPS'17, pp. 3646–3655. Red Hook: Curran Associates Inc.
42. Peysakhovich, A., & Lerer, A. (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, pp. 2043–2044. International Foundation for Autonomous Agents and Multiagent Systems.

43. Phan, T., Ritz, F., Belzner, L., Altmann, P., Gabor, T., & Linnhoff- Popien, C. (2021). VAST: Value function factorization with variable agent sub-teams. In *Advances in neural information processing systems*, pp. 24018–24032. Curran Associates Inc.
44. Phan, T., Sommer, F., Altmann, P., Ritz, F., Belzner, L., & Linnhoff-Popien, C. (2022). Emergent cooperation from mutual acknowledgment exchange. In *21st international conference on autonomous agents and multiagent systems (AAMAS)*, AAMAS '22, pp. 1047–1055. International Foundation for Autonomous Agents and Multiagent Systems.
45. Radke, D., Larson, K., Brecht, T., & Tilbury, K. (2023). Towards a better understanding of learning with multiagent teams. In *Proceedings of the 32nd international joint conference on artificial intelligence, IJCAI-23*, pp. 271–279. International Joint Conferences on Artificial Intelligence Organization, 8.
46. Rapoport, A. (1974). Prisoner's dilemma - recollections and observations. In *Game theory as a theory of a conflict resolution*, pp. 17–34. Springer.
47. Rapoport, A., Chammah, A. M., & Orwant, C. J. (1965). *Prisoner's dilemma: A study in conflict and cooperation*, volume 165. University of Michigan Press.
48. Ritz, F., Ratke, D., Phan, T., Belzner, L., & Linnhoff-Popien, C. (2021). A sustainable ecosystem through emergent cooperation in multi-agent reinforcement learning. volume ALIFE 2021: The 2021 Conference on Artificial Life of *ALIFE 2021: The 2021 Conference on Artificial Life*. MIT Press, 07.
49. Roesch, S., Leonardos, S., & Du, Y. (2024). The selfishness level of social dilemmas. In *23rd international conference on autonomous agents and multiagent systems (AAMAS)*, AAMAS '24, pp. 2441–2443. International Foundation for Autonomous Agents and Multiagent Systems.
50. Schenato, L., & Gamba, G. (2007). A distributed consensus protocol for clock synchronization in wireless sensor network. *Proceedings of the 46th IEEE conference on decision and control*, pp. 2289–2294. IEEE.
51. Schmid, K., Belzner, L., Müller, R., Tochtermann, J., & Linnhoff-Popien, C. (2021). Stochastic market games. In: Zhi-Hua, Z. (ed.) *Proceedings of the thirtieth international joint conference on artificial intelligence, IJCAI-21*, pp. 384–390. International Joint Conferences on Artificial Intelligence Organization, 8.
52. Schmid, K., Belzner, L., Gabor, T., & Phan, T. (2018). Action markets in deep multi-agent reinforcement learning. In *Proceedings of the international conference on artificial neural networks*, pp. 240–249. Springer International Publishing.
53. Shalev-Shwartz, S., Shammah, S., & Shashua, A. (2016). Safe multi-agent reinforcement learning for autonomous driving. arXiv preprint arXiv:1610.03295.
54. Silver, D., Singh, S., Precup, D., & Sutton, R. S. (2021). Reward is enough. *Artificial Intelligence, 299*, 103535.
55. Speranzon, A., Fischione, C., & Johansson, K. H. (2006). Distributed and collaborative estimation over wireless sensor networks. In: *Proceedings of the 45th IEEE conference on decision and control*, pp. 1025–1030. IEEE.
56. Stone, P., Kaminka, G., Kraus, S., & Rosenschein, J. (2010). Ad hoc autonomous agent teams: collaboration without pre-coordination. In *Proceedings of the AAAI conference on artificial intelligence, 24*(1), 1504–1509.
57. Sutton, R. S. (1988). Learning to predict by the methods of temporal differences. *Machine Learning, 3*(1), 9–44.
58. Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT Press.
59. Sutton, R. S., McAllester, D. A., Singh, S. P., & Mansour, Y. (2000). Policy gradient methods for reinforcement learning with function approximation. In S. Solla, T. Leen, and K. Müller, (Eds.), *Advances in neural information processing systems*, vol. 12, pp. 1057–1063. MIT Press.
60. Tan, M. (1993). Multi-agent reinforcement learning: Independent versus cooperative agents. In *Proceedings of the tenth international conference on international conference on machine learning*, pp. 330–337. Morgan Kaufmann Publishers Inc.
61. Andrew, S. (2007). *Tanenbaum and Maarten Van Steen*. Distributed Systems: Principles and Paradigms. Prentice-Hall.
62. Trivers, R. L. (1971). The evolution of reciprocal altruism. *The Quarterly Review of Biology, 46*(1), 35–57.
63. Van Lange, P. A. M., Joireman, J., Parks, C. D., & Van Dijk, E. (2013). The psychology of social dilemmas: A review. *Organizational Behavior and Human Decision Processes, 120*(2), 125–141.
64. Vinitsky, E., Köster, R., Agapiou, J. P., Duéñez-Guzmán, E., Vezhnevets, A. S., & Leibo, J. Z. (2021). A learning agent that acquires social norms from public sanctions in decentralized multi-agent settings. arXiv preprint arXiv:2106.09012.

65. Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., et al. (2019). Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature, 575*(7782), 350–354.
66. Wei, E., & Luke, S. (2016). Lenient learning in independent-learner stochastic cooperative games. *The Journal of Machine Learning Research, 17*(1), 2914–2955.
67. Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning, 8*(3), 229–256.
68. Yang, J., Li, A., Farajtabar, M., Sunehag, P., Hughes, E., & Zha, H. (2020). Learning to incentivize other learning agents. *Advances in Neural Information Processing Systems*, 33.
69. Yang, Y., Luo, R., Li, M., Zhou, M., Zhang, W., & Wang, J. (2018). Mean field multi-agent reinforcement learning. In *35th international conference on machine learning, ICML 2018*, vol. 80, pp. 5571–5580. PMLR.