Emergent Cooperation from Mutual Acknowledgment Exchange in Multi-Agent Reinforcement Learning

Thomy Phan (thomy.phan@ifi.lmu.de)
LMU Munich
Felix Sommer (felixissommer@hotmail.com)
LMU Munich
Fabian Ritz (fabian.ritz@ifi.lmu.de)
LMU Munich
Philipp Altmann (philipp.altmann@ifi.lmu.de)
LMU Munich
Jonas Nüßlein (jonas.nuesslein@ifi.lmu.de)
LMU Munich
Michael Kölle (michael.koelle@ifi.lmu.de)
LMU Munich
Lenz Belzner (lenz.belzner@thi.de)
Technische Hochschule Ingolstadt
Claudia Linnhoff-Popien (linnhoff@ifi.lmu.de)
LMU Munich

Research Article

Keywords:

DOI: https://doi.org/

Emergent Cooperation from Mutual Acknowledgment Exchange in Multi-Agent Reinforcement Learning

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

[1]LMU Munich, Germany.
[2]Technische Hochschule Ingolstadt, Germany.

*Corresponding author(s). E-mail(s): thomy.phan@ifi.lmu.de;

Abstract

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

1 Introduction

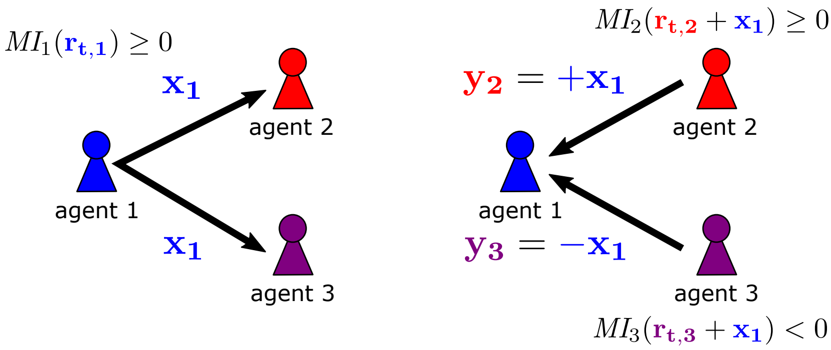
Many potential AI scenarios like autonomous driving [38], smart grids [11], or general IoT scenarios [8], where multiple autonomous systems coexist within a shared environment, can be naturally modeled as self-interested *multi-agent system (MAS)* [26, 6]. In self-interested MAS, each autonomous system or agent attempts to achieve an individual goal while adapting to its environment, i.e., other agents' behavior [13]. Conflict and competition are common in such systems due to opposing goals or shared resources [26, 31].

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 *social dilemma (SD)* [34]. SDs can be temporally extended to *sequential social dilemmas (SSD)* to model more realistic scenarios [23].

Multi-agent reinforcement learning (MARL) has become popular to model individually rational agents in SDs and SSDs to examine emergent behavior [6, 23, 31, 15, 36]. The goal of each agent is defined by an individual reward function. Non-cooperative game theory and empirical studies have shown that naive MARL approaches commonly fail to learn cooperative behavior due to individual selfishness and lacking benevolence towards other agents, which leads to defective behavior [2, 47, 23, 13].

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 [44, 6, 22, 17]. This can cause agents to defect from mutual cooperation as studied extensively for the prisoner's dilemma [34, 2, 23, 13]. To mitigate this problem, some approaches propose to adapt the learning rate based on the outcome [5, 30, 50] or to integrate information of other agents' adaptation like gradients or opponent models [13, 25, 21]. These approaches are either tabular or require full observability in order to observe each other's behavior thus do not scale to complex domains. Furthermore, some approaches require knowledge about other agents' objective to estimate their degree of adaptation which could violate privacy [13, 25].

Another reason for mutual defection is the *reward structure* which was found to be crucial for social intelligence [23, 39]. Prior work has shown that adequate reward formulations can lead to emergent cooperation in particular domains [3, 9, 10, 32, 19]. 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 [18, 20, 21, 52]. *Peer incentivization (PI)* is a distributed approach, where all agents learn to reward or to penalize each other which often leads to emergent cooperation [29, 52, 37, 48]. Current PI mechanisms implicitly assume a flawless communication channel in order to exchange rewards. These rewards are assumed to be simply integrated into the learning process without any chance to respond with feedback. Furthermore, most PI approaches rely on global information like joint actions [52], a central market function [37], or publicly available information [48] which limits scalability and applicability to real-world scenarios, where only local information is accessible.



(a) Request phase (b) Response phase

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

Once emergent cooperation has been achieved, it needs to be maintained to withstand social pressure, where many agents compete for scarce resources [23, 31], or disturbances like protocol defections or communication failures [2, 7]. Thus, *reciprocity* is important to establish stable cooperation by adequately responding to both cooperative and defective opponent behavior [35, 2, 1]. While reciprocity has already been considered in some prior learning rules [27, 5, 13, 25], 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 [29, 52, 37]. The lack of reciprocity on the reward-level can therefore lead to naive cooperation which can be easily destabilized.

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

- The concept of monotonic improvement, where each agent locally evaluates its individual situation to estimate the reliability of the environment, i.e., other agents' behavior.
- The MATE communication protocol and reward formulation using monotonic improvement estimation. The two phases of MATE ensure reward-level reciprocity, where 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 defect from the protocol and where 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 [33], 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 as well as additional experiments examining the sensitivity of MATE w.r.t. the choice of token values.

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* $\tau_{t,i} \in (\mathcal{Z} \times \mathcal{A}_i)^t$. $\pi_i(a_{t,i}|\tau_{t,i})$ is the action selection probability represented by the individual *policy* of agent i. In addition, we assume each agent i to have a *neighborhood* $\mathcal{N}_{t,i} \subseteq \mathcal{D} - \{i\}$ of other agents at every time step t which is domain dependent as suggested in [53].

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 [23, 31, 20, 29].

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 [35, 2, 47]. In SDs and SSDs, nash equilibria do not maximize efficiency (U) of a MAS therefore individually rational agents may fail to learn cooperative behavior [2, 1, 7, 23, 13].

2.2 Multi-Agent Reinforcement Learning

We focus on decentralized or independent learning, where each agent i optimizes its policy π_i based on local information like $\tau_{t,i}$, $a_{t,i}$, $r_{t,i}$, $z_{t+1,i}$ (and optionally information obtained from its neighborhood $\mathcal{N}_{t,i}$) using *reinforcement learning (RL)* techniques, e.g., policy gradient methods as explained in Section 2.3 [44, 13, 53]. *Naive (independent) learning* induces non-stationarity due to simultaneously adapting agents which continuously changes the environment dynamics [26, 22, 17]. Therefore, naive learning can lead to overly greedy and exploitative policies which defect from any cooperative behavior [23, 13].

2.3 Policy Gradient Reinforcement Learning

Policy gradient RL is a popular approach to approximate best responses π_i^* for each agent i [28, 13, 52]. 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 $J = \mathbb{E}_\pi[G_{0,i}]$ [51]. Most policy gradient methods use gradients g of the following form [43]:

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

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 [13]. 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 [44, 26, 6, 49]. Different studies have been conducted on various complex SSDs, where interesting phenomena like group hunting, attacking and dodging, or flocking have been observed [23, 31, 15, 36]. Independent MARL like naive learning has been widely used in most studies to model agents with individual rationality [44, 13].

3.2 Non-Stationarity in Multi-Agent Reinforcement Learning

Non-stationarity is one reason why naively learning agents fail to cooperate in SDs [44, 26, 6, 22, 17]. To mitigate this issue, different learning rates can be used depending on the outcome [5, 30, 50]. Another approach is to integrate "opponent awareness" into the learning rule by using or approximating other agents' gradients [13, 25]. 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 [29], 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 [52]. 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 [37], where the action space is extended by joint market actions to enable bilateral agreements between agents. A central market function is required which redistributes rewards depending on selling-buying relationships. This approach is intractable for large and complex scenarios because of the exponential growth of the individual action space, since each agent has to additionally decide on a joint market action. Furthermore, this approach does not enable penalization of agents. Another approach based on public sanctioning has been proposed in [48]. 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 by adequately responding to other agents' actions [35, 2, 1, 7]. *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 action of the other agent [35]. TFT is able to achieve and maintain emergent cooperation in simple games like the iterated prisoner's dilemma while being able to defend itself against exploitation based on the following characteristics [2, 1]:

- **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 [46]. 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 [23, 31].

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 [29, 52, 37] and independent adaptation of $\hat{\pi}_i$ and $\hat{V}_i$ [23, 31, 13].

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
            $e_{t,i} \leftarrow \langle \tau_{t,i}, a_{t,i}, r_{t,i}, z_{t+1,i} \rangle$
            $\hat{r}^{MATE}_{t,i} \leftarrow MATE(MI_i, \hat{V}_i, \mathcal{N}_{t,i}, \tau_{t,i}, e_{t,i})$ (See Algorithm 2)
            $e_{t,i} \leftarrow \langle \tau_{t,i}, a_{t,i}, \hat{r}^{MATE}_{t,i}, z_{t+1,i} \rangle$
            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 evaluate the *monotonic improvement* of their individual situation with a metric $MI_{e_{t,i},\hat{V}_i}$ or MI_i for short based on local information, i.e., rewards $r_{t,i}$, histories $\tau_{t,i}$, and messages exchanged with other agents $j \in \mathcal{N}_{t,i}$. Given some *arbitrary reward*

$\hat{r}_{t,i}$, which could be either the original reward $r_{t,i}$ or some shaped reward, agent i can assume a monotonic improvement of its situation when $MI_i(\hat{r}_{t,i}) \geq 0$. Note that we consider the case of $MI_i(\hat{r}_{t,i}) = 0$ as monotonic improvement in particular to encourage agents to maintain their cooperative behavior instead of falling back to defective strategies.

MI_i represents a heuristic local reliability measure to predict if an agent i can rely on its environment represented by other agents $j \in \mathcal{N}_{t,i}$ without loosing performance. Since MI_i can be measured online, agent i is able to 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* improvement of agent i, i.e., how $\hat{r}_{t,i}$ compares to all rewards obtained so far.

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

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

which corresponds to the TD residual w.r.t. to some *arbitrary reward* $\hat{r}_{t,i}$ and estimates the expected *long-term* improvement of agent i, i.e., how $\hat{r}_{t,i}$ and $\tau_{t+1,i}$ improve or degrade the situation of agent i w.r.t. future time steps [41, 42].

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

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

The complete formulation of MATE at time step t for any agent i is given in Algorithm 2. MI_i is a metric for estimating the individual monotonic improvement, $\hat{V}_i$ is the approximated value function, $\mathcal{N}_{t,i}$ is the current neighborhood, $\tau_{t,i}$ is the history, and $e_{t,i}$ is the experience tuple obtained at time step t. MATE computes and returns the shaped reward $\hat{r}_{t,i}^{MATE}$ (Eq. 4.2), 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 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 [28, 13, 25, 37, 48]. Locality of information is more practicable in real-world scenarios as global communication is typically expensive or infeasible, and disturbances mainly occur locally therefore should not affect the whole MAS [45]. 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 $r_{t,i}$, values $\hat{V}_i(\tau_{t,i})$, or TD residuals. This can be useful for open scenarios like adhoc teamwork or IoT settings, where arbitrary agents can join the system without revealing any private information or depending on central instances [4, 40]. 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 [16], MATE is able to respond immediately using rewards or penalties. Therefore, MATE exhibits the characteristics listed in Section 3.5 given that all agents use $\hat{r}_{t,i}^{MATE}$ according to Eq. 4.2 for adaptation:

- **Niceness**: The request phase of MATE only uses positive rewards $x_{token} > 0$ thus never defects first on 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. 4.2.
- **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 Section 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 [23, 31]. 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 [29]. 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 Section 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 [52] is left for future work. We note that agent-wise adaption of x_{token} might affect clarity according to Section 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 [37]. 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.

5 Experimental Setup

5.1 Evaluation Domains

We implemented three SD domains based on previous work [13, 31, 29]. 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. 2.1. Further details are in the 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 initial state s_0. One 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 $j \neq i$.

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 [24, 13]. If an agent collects a coin, it receives a reward of +1. However, if the coin has a different color than the collecting agent, another agent with the actual matching color is penalized with -2. After being collected, the coin respawns randomly with a new random color. All agents can observe the whole field and are able to move north, south, west, and east. An agent is only able to determine if a coin has the same or a different color than itself, but it is unable to distinguish anything further between colors. An episode terminates after 150 time steps and we set $\gamma = 0.95$. The neighborhood $\mathcal{N}_{t,i} = \mathcal{D} - \{i\}$ is defined by all other agents $j \neq i$. In addition to the efficiency, we measure the *"own coin" rate* $P(own\ coin) = \frac{\#\ collected\ coins\ with\ same\ color}{\#\ all\ collected\ coins}$ based on the coins collected by each agent.

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 [31]. 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 [31]:

$$E = 1 - \frac{\sum_{i \in \mathcal{D}} \sum_{j \in \mathcal{D}} |R_i - R_j|}{2N \sum_{i \in \mathcal{D}} R_i},$$
$$S = \frac{1}{N} \sum_{i \in \mathcal{D}} \Delta_i, \text{ where } \Delta_i = \mathbb{E}[t | r_{t,i} > 0],$$
$$P = N - \frac{1}{T} \sum_{i \in \mathcal{D}} \sum_{t=1}^{T} \mathbb{I}[\text{agent timed-out on time step } t]$$

5.2 MARL Algorithms

We implemented MATE as specified in Algorithm 2 with MI_i^{TD} (Eq. 4.1) and MI_i^{rew} (Eq. 4.1), 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. 2.3, which we refer to as *Naive Learning* [13].

In addition, we implemented *LIO* [52], the zero-sum and replenishable budget version of *Gifting* [29], 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 [13] 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 *multilayer perceptron (MLP).* Since *Coin[N]* and *Harvest[N]* are gridworlds, states and observations are encoded as multi-channel image as proposed in [14, 23]. The observations of *IPD* are the vector-encoded joint actions of the previous time step [13]. 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 B1 and further details are in the Appendix B.

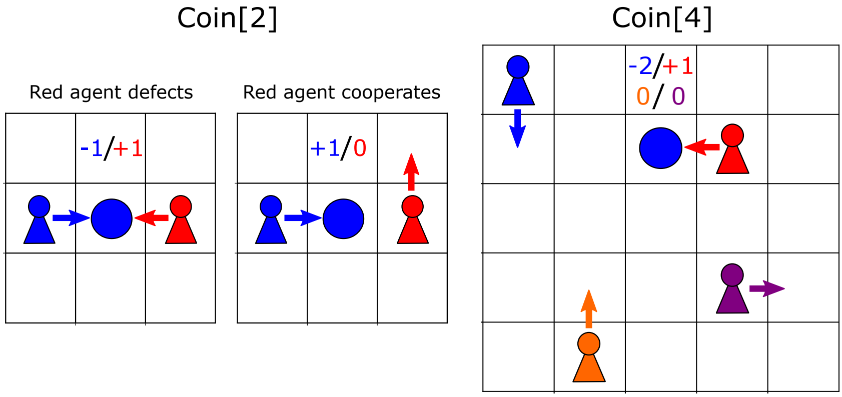
6 Results

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.

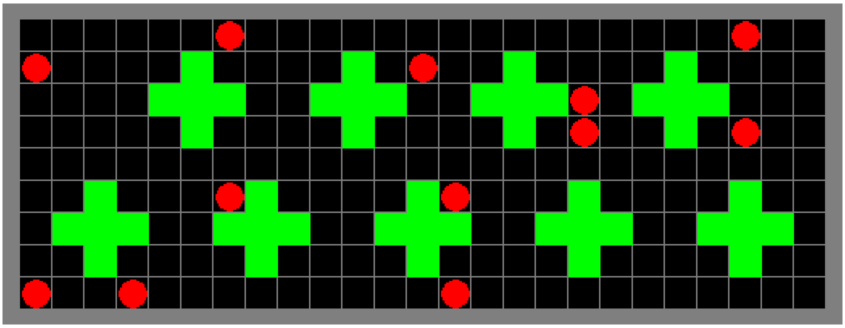
6.1 Performance Evaluation

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



(a) Coin



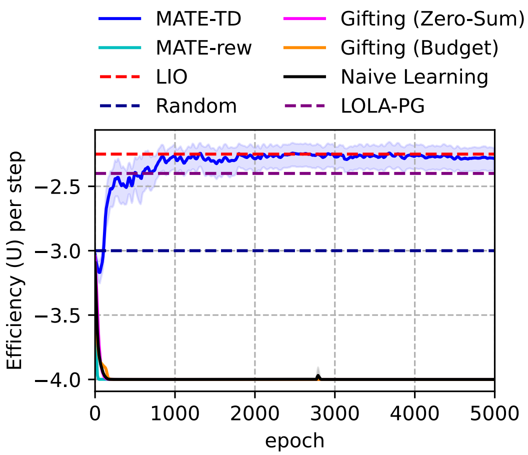
(b) Harvest (layout used for $N = 6$ and $N = 12$)

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.

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]* (after *Random*), but its "own coin" rate is similar to *Random*. Both *Gifting* variants and *Naive Learning* perform similarly to *Random* in *Coin[2]* but are significantly less efficient than *Random* in *Coin[4]*.

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

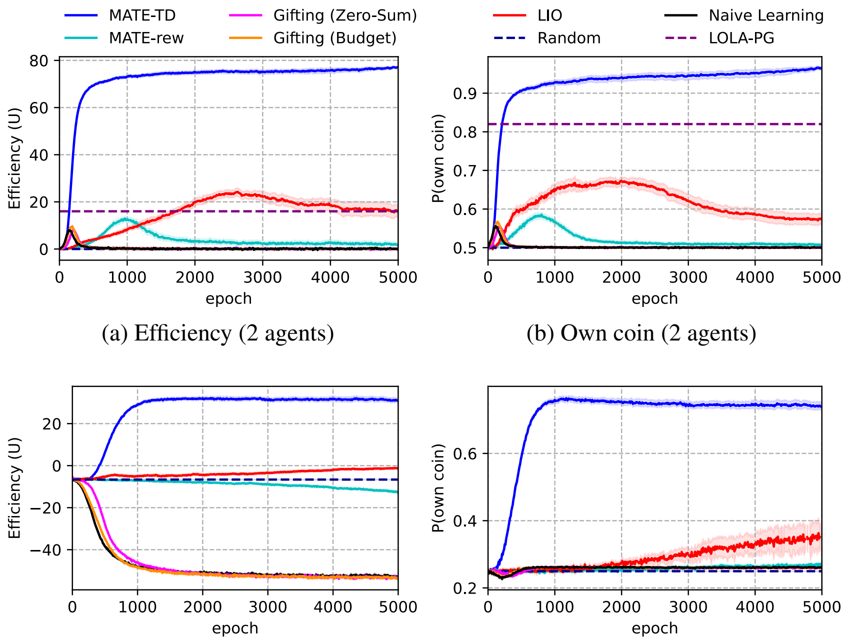
(a) Prisoner's Dilemma payoffs



(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 [52, 13].

The results for *Harvest[6]* and *Harvest[12]* are shown in Fig. 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 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 slowlier than *Gifting* and *Naive Learning*, but eventually surpasses them w.r.t. efficiency, sustainability, and peace. *MATE-rew* is the least efficient and sustainable MARL approach which

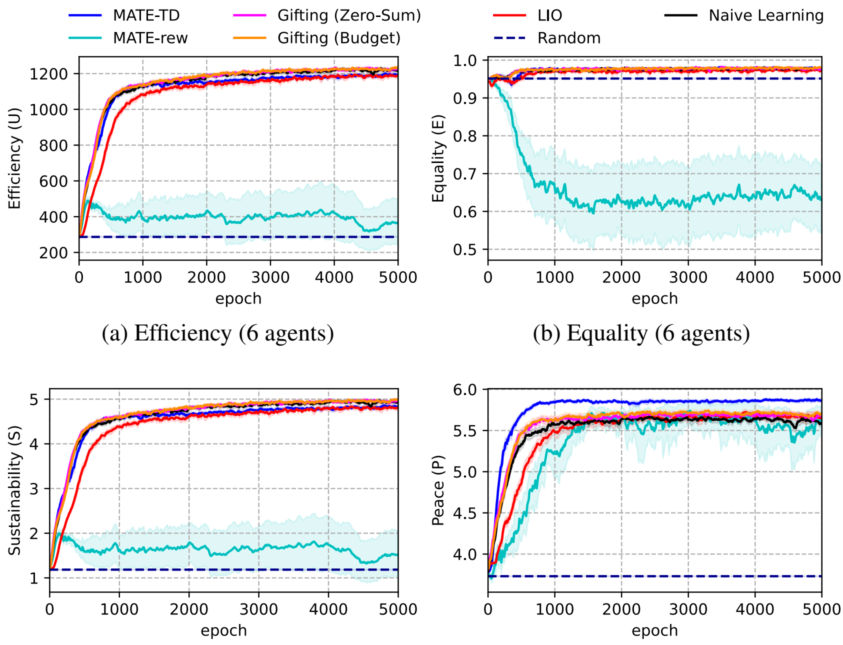


(a) Efficiency (2 agents) (b) Own coin (2 agents)

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

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 [13].

exhibits significantly less equality than *Random*. *LIO*, both *Gifting* variants, and *Naive Learning* first improve w.r.t. to 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 *tragedy of the commons* [31, 29]. However, *MATE-TD* remains stable w.r.t. efficiency, equality, and sustainability in *Harvest[12]*.



(a) Efficiency (6 agents)

(b) Equality (6 agents)

(c) Sustainability (6 agents)

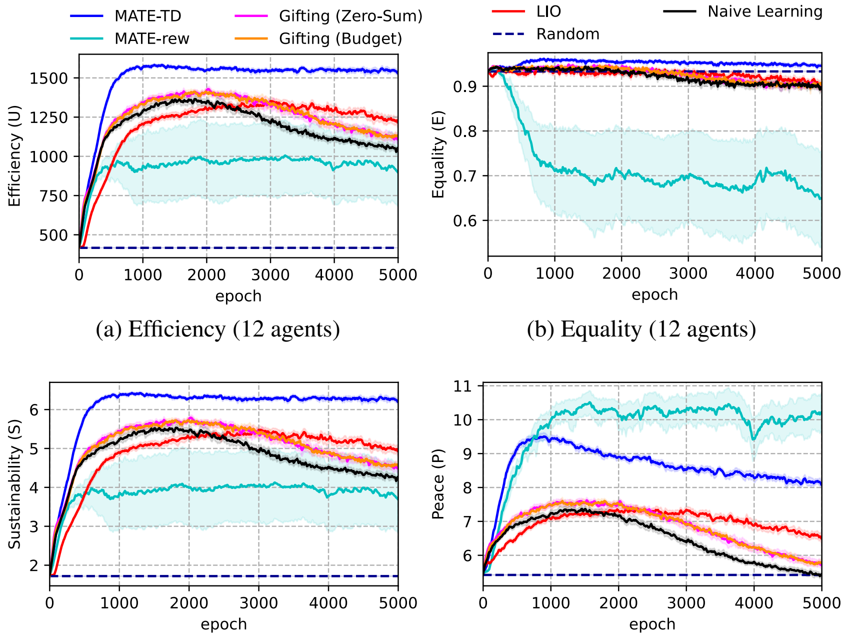
(d) Peace (6 agents)

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

6.2 Robustness against Protocol Defections

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

- **Complete**: The defector becomes a naive independent learner which does not participate in the communication rounds by skipping line 16 and 17 in Algorithm 1. Thus, the defector f simply learns with its original reward $r_{t,f}$. This defection strategy lacks niceness, retaliation, and forgiveness according to Section 4.3.2.

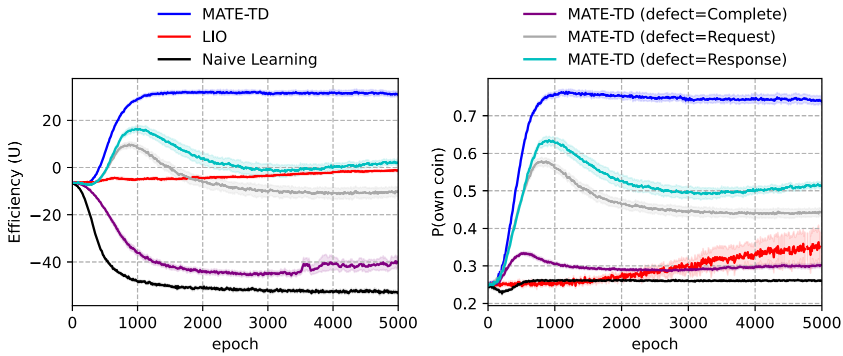


(a) Efficiency (12 agents) (b) Equality (12 agents)

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

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

- **Request**: The defector f does not send any acknowledgment requests by skipping line 4 in Algorithm 2 and receives no responses in return. However, it can still receive requests from other agents $j \in \mathcal{N}_{t,f}$ and respond to them. Thus, the defector'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 defection strategy lacks niceness according to Section 4.3.2.
- **Response**: The defector f can send acknowledgment requests but ignores all responses by skipping line 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 defector's reward $\hat{r}_{t,f}^{MATE}$ is the same as in the *Request* case above. This defection



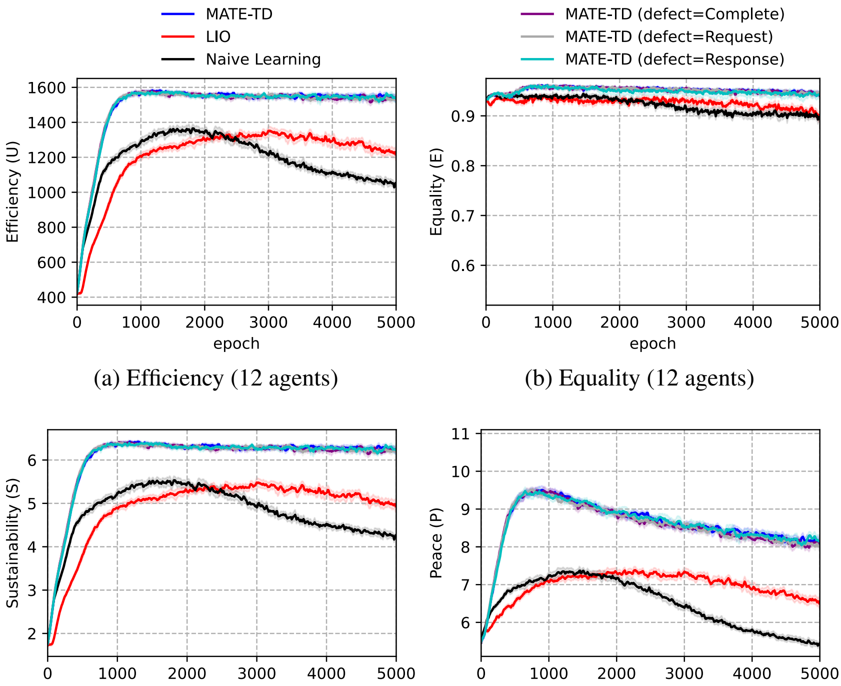
(a) Efficiency (4 agents) (b) Own coin (4 agents)

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

strategy does not lack any characteristic discussed in Section 4.3.2. However, the defector does not adapt its policy with the original MATE reward defined in Eq. 4.2.

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

The results for *Coin[4]* are shown in Fig. 7. All defective *MATE-TD* variants are less efficient than *MATE-TD* but still more efficient with a higher "own coin" rate than *Naive Learning*. *MATE-TD (defect=Complete)* exhibits the least degree of cooperation. *MATE-TD (defect=Response)* is slightly more efficient than *LIO* and achieves a higher "own coin" rate. *MATE-TD (defect=Request)* is less efficient than *LIO* but its "own coin" rate is higher indicating that agents tend to refrain from collecting other agents' coins rather than greedily collecting them.



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

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

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

The results for *Harvest[12]* are shown in Fig. 8. All defective *MATE-TD* variants perform similarly to *MATE-TD* without any loss.

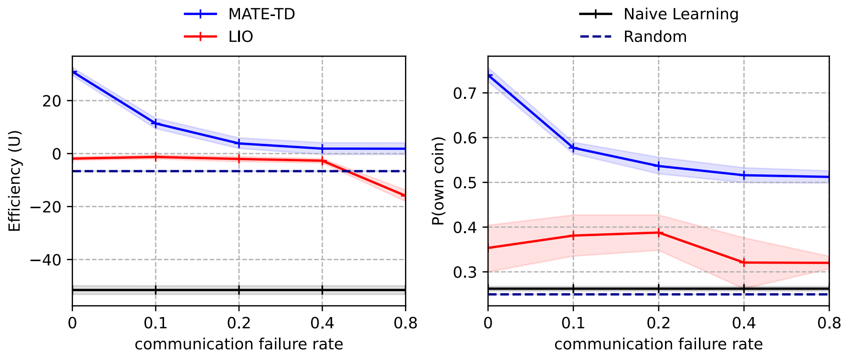
6.3 Robustness against Communication Failures

To evaluate robustness against communication failures, we introduce a *failure rate* $\delta \in [0, 1)$ specifying that an agent can fail to send or receive a message with a probability of δ. E.g., in the request phase in Fig. 1a, agent 1 could fail to send any request by skipping line 4 in Algorithm 2 with a probability of δ. If the requests are sent successfully, agent 2 or 3 can still fail at receiving agent 1's request by skipping lines 7-14 in Algorithm 2 with a probability of δ. The response phase in Fig. 1b is modeled analogously.

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 Section 5.1, communication in *Coin[4]* is *global*, where all-to-all communication is possible, while communication in *Harvest[12]* is *local* for *MATE-TD*, where all agents can 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 [52]. In addition, we compare to *Naive Learning* and *Random* as non-communicating baselines.

The results for *Coin[4]* are shown in Fig. 9. *MATE-TD* and *LIO* remain more efficient and cooperative than *Naive Learning* despite both approaches loosing performance with increasing δ. The average efficiency of *MATE-TD* is always nonnegative, while the efficiency of *LIO* decreases below the level of *Random*, when $\delta = 0.8$. The average "own coin" rate of *MATE-TD* is always at least 0.5, while the average "own coin" rate of *LIO* has a high variance ranging from 0.3 to 0.4. However, when $\delta = 0.8$, the average "own coin" rate of *LIO* is slightly above 0.3 with significantly less variance, while still being higher than the "own coin" rates of *Naive Learning* and *Random*.

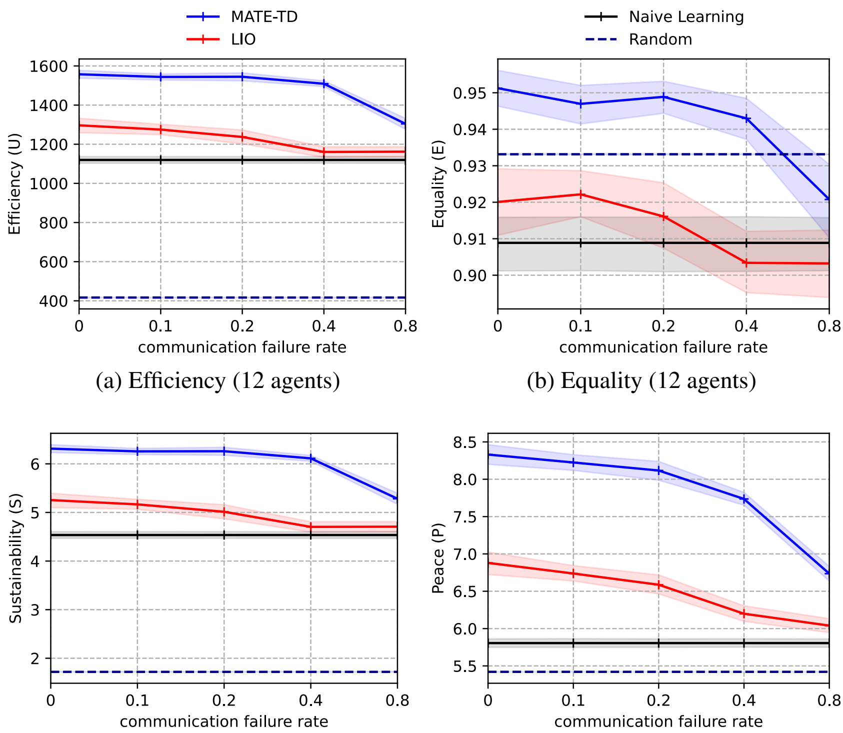
The results for *Harvest[12]* are shown in Fig. 10. The performance of *MATE-TD* is relatively robust for $\delta \geq 0.4$ but significantly drops when $\delta = 0.8$. However, *MATE-TD* still achieves the highest degree of cooperation w.r.t. all metrics except equality which gets worse than *Random* 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 $\delta \leq 0.4$.



(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.



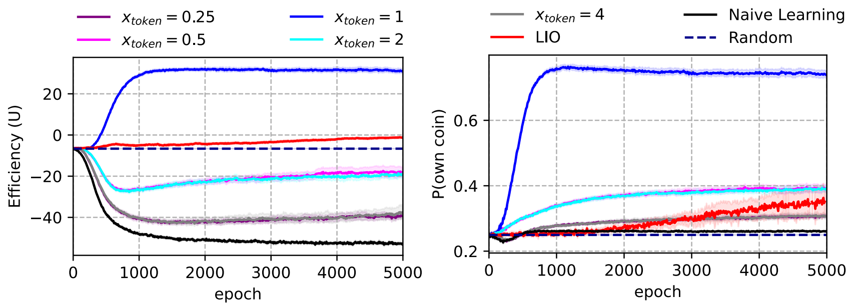
(a) Efficiency (12 agents)

(b) Equality (12 agents)

(c) Sustainability (12 agents)

(d) Peace (12 agents)

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



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

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]*.

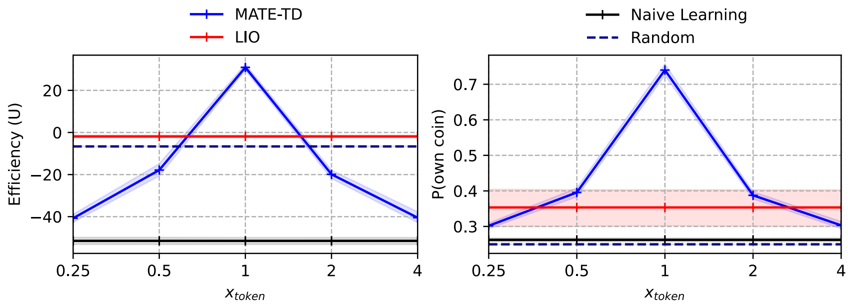
6.4 Sensitivity to Token Values

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 Section 5.1.

The results for *Coin[4]* are shown in Fig. 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 Fig. 13 and 14. All *MATE-TD* variants progress stably w.r.t. efficiency and sustainability without any gradual decrease. *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



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

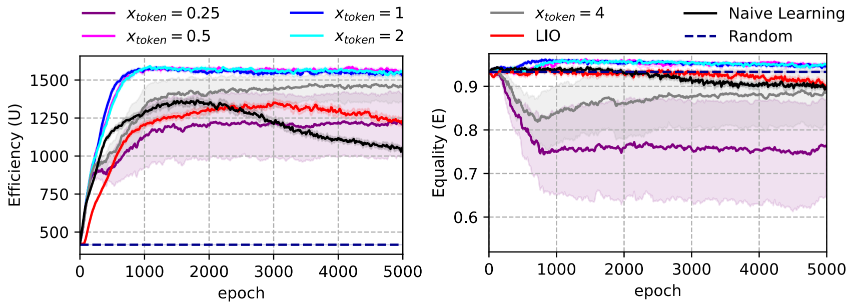
Fig. 12: Performance of MATE with $x_{token} \in \{0.25, 0.5, 1, 2, 4\}$, LIO, Naive Learning, and Random in *Coin[4]* after 5,000 epochs.

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.

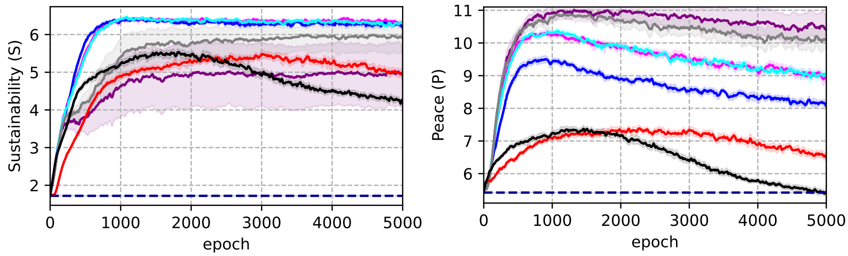
7 Discussion

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 despite the increased social pressure compared to *Harvest[6]*, where all alternative PI approaches easily learn to cooperate.

Estimating the monotonic short-term improvement via MI_i^{rew} (Eq. 4.1) can be beneficial compared to random acting and to some extent to naive learning in Coin (Fig. 4). However, considering the monotonic long-term improvement via MI_i^{TD} (Eq. 4.1) leads to significantly higher efficiency and cooperation w.r.t. various metrics in all domains, except peace in *Harvest[12]*. MATE with



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

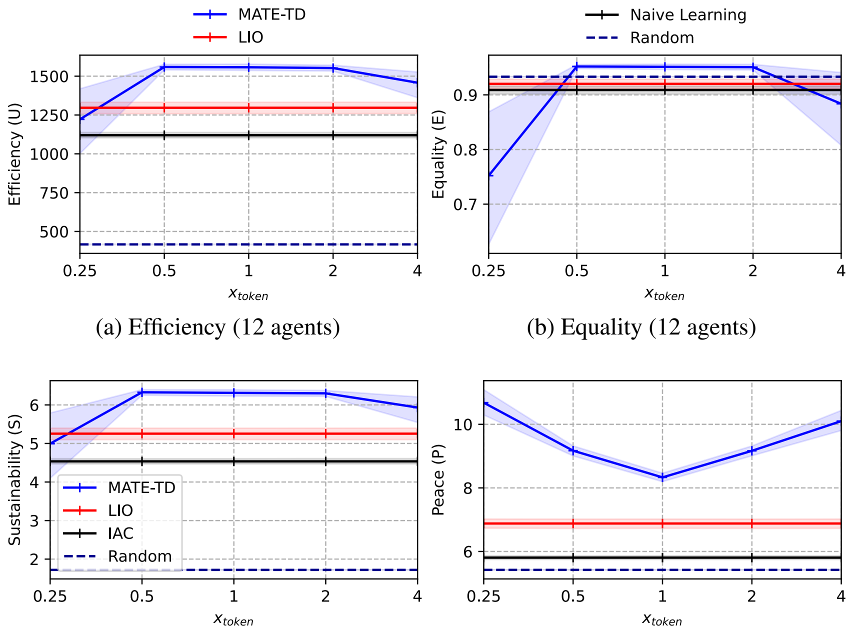


(c) Sustainability (12 agents) (d) Peace (12 agents)

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]*.

MI_i^{TD} is able to maintain cooperative behavior, in contrast to other approaches which become unstable and fall back to more defective strategies as observed in *Coin[2]*, *Coin[4]*, and *Harvest[12]* (Fig. 4 and 6).

MATE is not affected by single protocol defectors in *Harvest[12]*, while its cooperation level significantly decreases in *Coin[4]*, where any deviation from the protocol can affect the whole MAS (Fig. 7 and 8). The protocol defection in *Coin[4]* emphasizes the importance of appropriate penalization mechanisms as proposed in our reward formulation in Eq. 4.2 for immediate retaliation according to Section 4.3.2 and [2, 1, 7]. Niceness through initiation of the MATE protocol according to Section 3.5 is also important as MATE defectors using the strategy *Response* lead to superior cooperation in *Coin[4]* than defectors using *Request*. Forgiveness is always implicitly assumed except for



(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.

the defection strategy *Complete* which leads to the least cooperative behavior in *Coin[4]*.

MATE shows some robustness against communication failures in Fig. 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 $\delta \geq 10\%$.

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 Section 4.3.3, setting x_{token} to the highest positive reward yields the best results w.r.t. most metrics as shown in Fig. 11-14 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 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.

8 Conclusion and Future Work

We presented MATE, a PI approach defined by a two-phase communication protocol to mutually exchange acknowledgment tokens to shape individual rewards. Each agent evaluates the monotonic improvement of its individual situation in order to accept or reject acknowledgment requests from other agents. MATE is completely decentralized and only requires local communication and information without knowledge about other agents' objective 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 protocol defections 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 degree at least.

MATE is suitable for more realistic scenarios, e.g., in adhoc 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 techniques to create more adaptive and intelligent agents with social capabilities [12, 39].

References

[1] Robert Axelrod. *The Evolution Of Cooperation.* Basic Books, 1984.

[2] Robert Axelrod and William D. Hamilton. The Evolution of Cooperation. *Science*, 211(4489):1390–1396, 1981.

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

[4] Samuel Barrett, Peter Stone, and Sarit Kraus. 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, page 567–574. International Foundation for Autonomous Agents and Multiagent Systems, 2011.

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

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

[7] Richard Dawkins. *The Selfish Gene: 40th Anniversary Edition*. Oxford Landmark Science. OUP Oxford, 2016.

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

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

Multiagent Systems, 2011.

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

[11] Aris L. Dimeas and Nikos D. Hatziargyriou. Multi-Agent Reinforcement Learning for Microgrids. In *IEEE PES General Meeting*, pages 1–8. IEEE, 2010.

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

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

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

[15] Carsten Hahn, Thomy Phan, Thomas Gabor, Lenz Belzner, and Claudia Linnhoff-Popien. Emergent Escape-Based Flocking Behavior using Multi-Agent Reinforcement Learning. volume ALIFE 2019: The 2019 Conference on Artificial Life of *ALIFE 2021: The 2021 Conference on Artificial Life*, pages 598–605. MIT Press, 07 2019.

[16] Daniel Hennes, Dustin Morrill, Shayegan Omidshafiei, Rémi Munos,

Julien Perolat, Marc Lanctot, Audrunas Gruslys, Jean-Baptiste Lespiau, Paavo Parmas, Edgar Duèñez Guzmán, and Karl Tuyls. Neural Replicator Dynamics: Multiagent Learning via Hedging Policy Gradients. In *Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems*, AAMAS '20, page 492–501. International Foundation for Autonomous Agents and Multiagent Systems, 2020.

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

[18] Edward Hughes, Joel Z. Leibo, Matthew Phillips, Karl Tuyls, Edgar Dueñez Guzman, Antonio García Castañeda, Iain Dunning, Tina Zhu, Kevin McKee, Raphael Koster, Heather Roff, and Thore Graepel. Inequity Aversion Improves Cooperation in Intertemporal Social Dilemmas. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

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

[20] Max Jaderberg, Wojciech M. Czarnecki, Iain Dunning, Luke Marris, Guy Lever, Antonio Garcia Castaneda, Charles Beattie, Neil C. Rabinowitz, Ari S. Morcos, Avraham Ruderman, et al. Human-Level Performance in 3D Multiplayer Games with Population-based Reinforcement Learning.

Science, 364(6443):859–865, 2019.

[21] Natasha Jaques, Angeliki Lazaridou, Edward Hughes, Caglar Gulcehre, Pedro Ortega, Dj Strouse, Joel Z. Leibo, and Nando De Freitas. Social Influence as Intrinsic Motivation for Multi-Agent Deep Reinforcement Learning. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 3040–3049. PMLR, 09–15 Jun 2019.

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

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

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

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

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

[27] Michael L. Littman. Friend-or-Foe Q-Learning in General-Sum Games. In *Proceedings of the Eighteenth International Conference on Machine*

Learning, ICML '01, page 322–328, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

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

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

[30] Laëtitia Matignon, Guillaume J. Laurent, and Nadine Le Fort-Piat. 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*, pages 64–69. IEEE, 2007.

[31] Julien Perolat, Joel Z. Leibo, Vinicius Zambaldi, Charles Beattie, Karl Tuyls, and Thore Graepel. 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, page 3646–3655, Red Hook, NY, USA, 2017. Curran Associates Inc.

[32] Alexander Peysakhovich and Adam Lerer. Prosocial Learning Agents Solve Generalized Stag Hunts Better than Selfish Ones. In *Proceedings of the 17th International Conference on Autonomous Agents and Multi-Agent Systems*, AAMAS '18, page 2043–2044. International Foundation for Autonomous Agents and Multiagent Systems, 2018.

[33] Thomy Phan, Felix Sommer, Philipp Altmann, Fabian Ritz, Lenz Belzner, and Claudia Linnhoff-Popien. Emergent Cooperation from Mutual Acknowledgment Exchange. In *21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, AAMAS '22, page 1047–1055. International Foundation for Autonomous Agents and Multiagent Systems, 2022.

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

[35] Anatol Rapoport, Albert M. Chammah, and Carol J. Orwant. *Prisoner's Dilemma: A Study in Conflict and Cooperation*, volume 165. University of Michigan Press, 1965.

[36] Fabian Ritz, Daniel Ratke, Thomy Phan, Lenz Belzner, and Claudia Linnhoff-Popien. 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 2021.

[37] Kyrill Schmid, Lenz Belzner, Robert Müller, Johannes Tochtermann, and Claudia Linnhoff-Popien. Stochastic Market Games. In Zhi-Hua Zhou, editor, *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 384–390. International Joint Conferences on Artificial Intelligence Organization, 8 2021.

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

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

[40] Peter Stone, Gal Kaminka, Sarit Kraus, and Jeffrey Rosenschein. Ad Hoc Autonomous Agent Teams: Collaboration without Pre-Coordination. *Proceedings of the AAAI Conference on Artificial Intelligence*, 24(1):1504–1509, Jul. 2010.

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

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

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

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

[45] Andrew S. Tanenbaum and Maarten Van Steen. *Distributed Systems: Principles and Paradigms*. Prentice-Hall, 2007.

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

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

[48] Eugene Vinitsky, Raphael Köster, John P. Agapiou, Edgar Duéñez-Guzmán, Alexander Sasha Vezhnevets, and Joel Z. Leibo. A Learning Agent that Acquires Social Norms from Public Sanctions in Decentralized

Multi-Agent Settings. *arXiv preprint arXiv:2106.09012*, 2021.

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

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

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

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

[53] Yaodong Yang, Rui Luo, Minne Li, Ming Zhou, Weinan Zhang, and Jun Wang. Mean Field Multi-Agent Reinforcement Learning. In *35th International Conference on Machine Learning, ICML 2018*, volume 80, pages 5571–5580. PMLR, 2018.

Declarations

Competing Interests

The authors have no conflicts of interest to declare that are relevant to the content of this article.

Authors' Contributions

T.P. and F.S. designed and implemented the concepts. T.P., F.S., P.A., J.N., and L.B. discussed the concepts. T.P., F.S., F.R., and L.B. provided and discussed related work. T.P., F.S., and F.R. designed and conducted the experiments. T.P., F.S., F.R., M.K., and C.L. discussed the results and visualized the data. All authors contributed to writing the manuscript.

Funding

No funding was received to assist with the preparation of this manuscript.

Availability of Data and Materials

Our code is available at https://github.com/thomyphan/emergent-cooperation.

Appendix A Evaluation Domain Details

A.1 IPD

An *IPD* episode consists of 150 iterations similar to [13]. 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 [13] in *Coin[2]* as shown in Fig. A1 with the same rules and reward functions. In addition, we extend the domain to 4 agents in *Coin[4]* (Fig. A1 right).

Since all agents are able to observe 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.

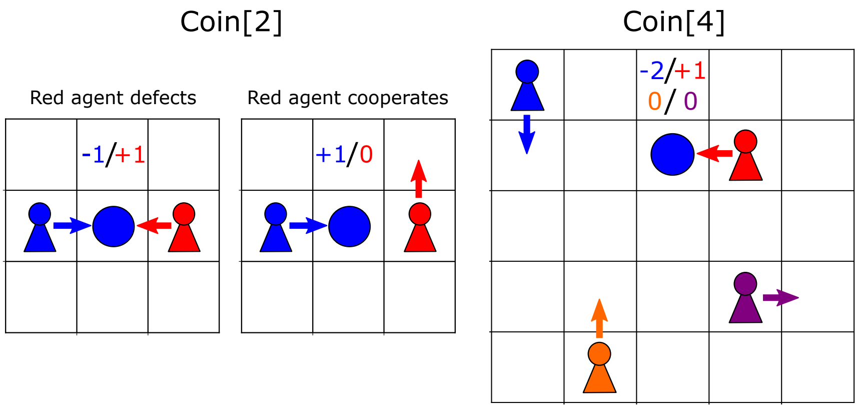


Fig. A1: *Coin[2]* and *Coin[4]* as used in the paper.

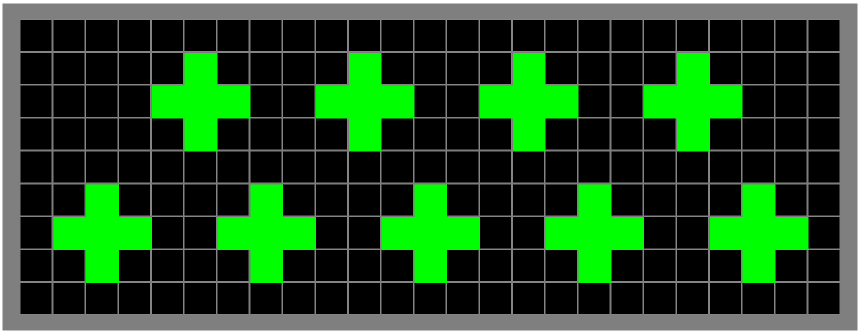


Fig. A2: Domain layout with initial apple configuration used for *Harvest[6]* and *Harvest[12]*.

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 simultaneously step on a coin.

A.3 Harvest[N]

We adopt the setup of [31] in *Harvest[6]* and *Harvest[12]* as shown in Fig. A2 with the same dynamics and apple regrowth rates. The initial apple configuration in Fig. A2 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 simultaneously attempt to collect an apple or tag each other.

Appendix B Technical Details

B.1 Hyperparameters

All common hyperparameters used by all MARL approaches in the experiments as reported in Section 6 are listed in Table B1. 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 B1 for all other approaches and domains from Section 5 and 6.

Similarly to $x_{token} = 1$, we set the gift reward of both *Gifting* variants introduced in Section 5.2 to 1 as originally proposed in [29].

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 [52].

B.2 Neural Network Architectures

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

Table B1: 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 Section B.2.
# units per hidden layer	64	$\{32, 64, 128\}$	Number of units per hidden layer. See Section B.2.
Hidden layer activation	ELU	{ReLU, ELU}	Activation function used for the hidden layer outputs. See Section 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	$\{1,\infty\}$	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[N]*) 0.99 (*Harvest[N]*)	$\{0.9, 0.95, 0.99\}$	Discount factor for the return $G_{t,i}$. Any value ≥ 0.95 would have been sufficient.
$\lvert\tau_{t,i}\rvert$	1	$\{1, 5, 10\}$	Local history length. It was set to 1 to reduce computation because the other values did not significantly improve performance.

Figures

Figure 1

Figure 2

Figure 3

Figure 4

Figure 5

Figure 6

Figure 7

Figure 8

Figure 9

Figure 10

Figure 11

Figure 12

Figure 13

Figure 14

Figure 15

Figure 16

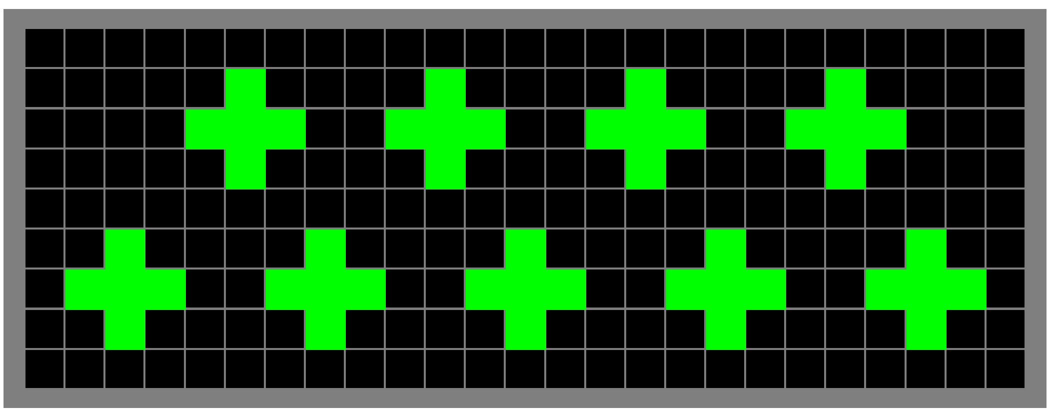


Figure 17