Counterfactual Multiagent Policy Gradients and Regret Minimization in Cooperative Settings

Chirag Chhablani,[1] **Ian Kash,** [1]
[1] Department of Computer Science
University of Illinois at Chicago
{cchhab2, iankash}@uic.edu

Abstract

Counterfactual Multi-agent Policy Gradients (COMA) is a popular algorithm for learning in cooperative multi-agent reinforcement learning settings where agent directly receive a global reward instead of an individual reward. COMA computes difference rewards to solve the multiagent credit assignment problem by providing a local learning signal for each agent. Despite good performance, there is a lack of theoretical justification for COMA's difference rewards. We provide such a justification by connecting COMA's update rule to regret minimization in identical interest games. This leads to two further insights. First, COMA's update rule may lead to slow policy updates even in very simple environments, and this can be ameliorated by a slight modification as previously observed in the Neural Replicator Dynamics algorithm. Second, this provides a justification for the use of a bounded softmax policy in terms of a guarantee of favorable convergence rates in identical interest games. Experimental results show the importance of these observations in more complex environments.

Cooperative multi-agent reinforcement learning (Coop-MARL) is a framework for many complex real-world reinforcement learning problems such as the coordination of autonomous vehicles (Cao et al. 2012), network packet delivery (Ye, Zhang, and Yang 2015), and distributed logistics (Ying and Dayong 2005). *Counterfactual Multi-Agent Policy Gradients (COMA)* is a recent technique for learning cooperation among agents. Key to COMA's success is efficient multiagent credit assignment through the implementation of *difference rewards* which were proposed by Wolpert and Tumer (2002) and Tumer and Agogino (2007). COMA uses difference rewards to evaluate the contribution of each agent's actions by comparing with the expected value of actions based on its current policy. For each agent, the difference reward signal represents the advantage of including the agent in the system compared to the counterfactual case when it is excluded from the system. This individual advantage signal is called the *counterfactual advantage baseline*. Despite good performance, there is not a theoretical justification for this particular choice of baseline. We argue that this baseline works well because it is similar to minimizing regret in a cooperative setting. Previously, regret minimization algorithms for stateful settings like counterfactual regret minimization (CFR) and its variants have primarily been studied in competitive settings like poker. Thus our analysis opens an opportunity to expand the advancements from Competitive MARL to Coop-MARL settings, which provides new structure that can be exploited.

Our theoretical results consider an identical interest game, which is a single-state version of shared reward Coop-MARL. This allows us to connect a variant of COMA's update rule to the classic regret minimization algorithm Hedge. There is a small literature on regret minimization algorithms in identical interest games showing how the structure allows stronger properties to be shown than in competitive settings. For example in some cases it is possible to show last iterate convergence to a pure Nash equilibrium (Mehta, Panageas, and Piliouras 2015), while in competitive settings it is typically only possible to show convergence on average to the weaker solution concept of (coarse) correlated equilibrium.

Beyond this high-level justification for COMA and emphasis of the importance of this direction for MARL research, we use this connection to make two further observations. First, we show that the current version of gradient update rule of COMA can lead to slow learning even in simple environments. This problem, and a solution to it known as Neural Replicator Dynamics (NRD) have previously been examined in competitive settings (Hennes et al. 2020). We argue that the structure of cooperative settings makes this fix less important for convergence, but still important for efficient learning. In our experimental results, we show that such an update rule leads to accelerated learning and higher performance in many game environments without adding any extra cost.

Second, we give a theoretical justification for COMA's use of a bounded softmax policy update instead of the usual softmax update. This update rule has been shown in recent work on potential games (a superset of identical interest games) to give faster convergence to pure Nash Equilibrium when updates are only applied to the action taken (Cohen, Héliou, and Mertikopoulos 2017). As this is the case for COMA, this connection provides an intuition for the good performance of this specific choice of policy representation. Our experimente provide evidence for this benefit in stateful settings, with some caveats.

Related Work

For Coop-MARL settings where agents receive a single global reward, decentralized learning leads to problem of multiagent credit assignment which is addressed by many techniques, including COMA, by generating a local reward for each agent. Such an approach typically performs better than directly using the global reward for each agent (Jianhong Wang et al. 2020). Various ways to assign local rewards from a single global reward have been proposed in previous work. Apart from COMA, Nguyen, Kumar, and Lau (2018) employ count-based variance reduction. Jianhong Wang et al. (2020) model such cooperative multiagent problems as extended convex game and propose the Shapley Q-value. Yang et al. (2018) perform credit assignment for more general case where individual agents have their own individual goals as well. Apart from using COMA-like advantages for obtaining, for each agent, they evaluate the advantages of action on other agent's individual goal. Other recent approaches include those taken by Zhou et al. (2020) and Son et al. (2020).

Regret minimization algorithms have been extensively studied in stateless multiagent settings and provide guarantees about convergence to Nash equilibrium in zero-sum games and (coarse) correlated equilibrium in general-sum games (Freund and Schapire 1999; Hart and Mas-Colell 2000). Their primary extension to stateful settings is the Counterfactual Regret Minimization (CFR) algoritm (Zinkevich et al. 2008), which provides similar rigorous guarantees under restrictive assumptions of perfect recall and terminal states. More recently a line of work has begun to explore connections between policy gradient approaches and regret minimization. Advantage Regret Minimization (Jin, Keutzer, and Levine 2018) drew an analogy between advantages and CFR's update rule and used it to demonstrate strong performance in POMDP settings. Srinivasan et al. (2018) explored different variants of actor critic algorithms that exploit this insight in stochastic games. In an approach we build on, Hennes et al. (2020) show how to draw an exact equivalence between regret minimization and policy gradient approaches by restricting to stateless settings. Recently, Yu et al. (2019) also use the idea of advantages for regret minimization in a cooperative setting, but they define team regret for group of agents and explore ways to divide the team regret for decentralized learning rather than directly doing regret minimization on the global reward.

Background

Cooperative Stochastic Games

COMA learns in a fully cooperative, global reward, partially observable, multi-agent environment that can be modelled as a stochastic game G, also known in this special case as a DEC-POMDP (Bernstein et al. 2002), defined by a tuple $G = (S, U, P, r, Z, O, n, \gamma)$. The number of agents is denoted by n, and we denote an arbitrary individual agent by $a \in A = \{1, ..., n\}$. The true state of the game is denoted by $s \in S$. At each time step t, every agent simultaneously chooses an action $u_a \in U$, forming a joint action $\mathbf{u} \in \mathbf{U} = U^n$. All the agents receive same global reward $r(s, \mathbf{u}) : S \times \mathbf{U} \rightarrow \mathbb{R}$ and the game transitions to a new state according to the transition function $P(s' \mid s, \mathbf{u})$. The goal of G is to maximize the expected discounted reward $R_t = \sum_{t=0}^{\infty} \gamma^t r_{t+1}$ where $\gamma \in [0, 1)$ is a discount factor. (In general quantities in bold will be used to denote joint variables and the notation $-a$ will used to denote variables for agents excluding agent a.) Further, we assume that from an agent's perspective, the environment is partially observable where the agents draw observations $z \in Z$ according to the observation function $O(s, a) : S \times A \rightarrow Z$. The agent maintains action-observation history $\tau_a \in T = (Z \times U)^* \times Z$, on which it conditions a stochastic policy $\pi^a(u_a|\tau_a) : T \times U \rightarrow [0, 1]$. COMA uses centralized training and decentralized execution (CTDE) learning framework to guide the training for each agent. In CTDE, the agents have access to the true state information and actions during the training phase while they only observe the local action-observation history during the actual execution in the environment.

COMA

COMA uses an actor-critic algorithm to maximize the discounted global reward for G and learn a policy for each agent. It uses a centralized critic with decentralized actor networks which implements the CTDE framework to update policy for each agent as shown in Algorithm 1. The input to the critic is the current state s_t and joint action $\mathbf{u}_t$ while the input to the actor network is the history h_t^a. The goal of the centralized critic is to estimate a value functions using sampled trajectories trajectories according to some current policy to learn the value functions $Q_\pi(s, \mathbf{u}) = \mathbb{E}_\pi[\sum_{k=t}^{\infty}[\gamma^{k-t} r_k | s_t = s, \mathbf{u}_t = \mathbf{u}]$ and $V_\pi(s) = \mathbb{E}_\pi[\sum_{k=t}^{\infty}[\gamma^{k-t} r_k | s_t = s]$.

Each agent plays an action u_t^a conditioned on agent's history h_t^a and sampling from current policy $\pi^a(u|h_t^a)$. The agents get the global reward r_t and reach the next state next state s_{t+1}. It uses this information to update the critic parameters θ^c by minimizing the mean square loss between the critic's target values y_t and critic values $Q(s, \mathbf{u})$. The critic's target network parameters are updated after every C steps. For updating the actor network, COMA uses difference rewards (Wolpert and Tumer 2002; Tumer and Agogino 2007) which measures the contribution of each agent to the global reward. COMA's equation for difference rewards is:

$$A^a(s, \mathbf{u}) = Q(s, \mathbf{u}) - \sum \pi^a(u'^a|\tau_a) Q(s, (u'^a, \mathbf{u}^{-a})) \quad (1)$$

The $A^a(s, u)$ is referred to as the *counterfactual baseline*. The first term on right hand side measures expected reward $Q(s, \mathbf{u})$ when agent a takes action u while second term is counterfactual scenario when a is removed from the system by subtracting its expected value based on current policy. This counterfactual baseline is used to update π^a (parameterized by θ_a^π) according to the standard policy gradient equation for actor-critic setting. For a given state-action pair $(s, \mathbf{u}))$ this is given by:

$$\theta_t = \theta_{t-1} + \eta_t \sum \nabla \log \pi_\theta^a(u^a|\tau^a; \theta_{t-1}) A^a(s, \mathbf{u}) \quad (2)$$

The centralized value network enables faster calculation of difference rewards as it reduces number of parameters for training and the number of forward passes required to estimate counterfactual advantage values while decentralized actor networks are useful in representing broad range of policies for different agents and allows the decentralized execution.

Algorithm 1 COMA update for agent **a**

```
Result: Update policy for agent $a$ give by $\pi^a$
Given the action of other agents $\mathbf{u}_t^{-a}$, play action $u_t^a$ based
  on agent's history $h_t^a$ for each agent $u_t^a$ and get reward $r_t$
  and next state $s_{t+1}$
  for t = 1 to T do
      Calculate the target network values $y_t$ using the critic
        parameter of the target $\hat{\theta_t^c}$ while batch unrolling states,
        actions and rewards.
end
for t = T to 1 do
    $\Delta\theta^c = \nabla_{\theta^c}(y_t - Q(s_t, \mathbf{u}))^2$
      $\Delta\theta^c \leftarrow \theta^c - \alpha\Delta\theta^c$
      Every C steps reset $\hat{\theta_t^c} = \theta_t^c$
end
for t = 1 to T do
    $A^a(s_t, \mathbf{u}) = Q(s_t, \mathbf{u}) - \sum_u Q(s_t, u, \mathbf{u}^{-a})\pi^a(u|h_t^a)$
      $\Delta\theta^q = \Delta\theta^a + \nabla_{\theta^a} \log \pi^a(u|h_t^a) A^a(s_t^a, \mathbf{u})$
end
$\theta_{t+1}^a = \theta_t^a + \alpha\Delta\theta^a$
```

Identical Interest Game

For our analysis of COMA, we use the special case where there is a single state, also known as an identical interest game in the game theory literature (Marden, Arslan, and Shamma 2007; Jianhong Wang et al. 2020). All identical interest games have at least one joint action $\mathbf{u}$ which is a maximizer of the shared reward $r(\mathbf{u})$. With such a joint action, indeed for any local optimum of r, for all agents a

$$r(u_a, \mathbf{u}_{-a}) \geq r(u_a, \mathbf{u}_{-a}). \tag{3}$$

Such a point is known as a pure Nash equilibrium.

Regret

Regret is a tool for for measuring the relative performance of a single action to a policy in an online setting. For an identical interest game, regret measures how much the agent's particular action led to a better global reward for the system while keeping the actions of other agents $\mathbf{u}^{-a}$ fixed.

$$Regret(a, \mathbf{u}) = r(\mathbf{u}) - \sum_u \pi^a r(u, \mathbf{u}^{-a}) \tag{4}$$

A key observation, not original to our work, is the structural similarity between Equations (1) and (4). The average regret

$$\frac{1}{T} Regret(a, u) \tag{5}$$

is minimized by an class of algorithms known as regret minimization, or no-regret, algorithms. The objective of no-regret algorithms is to reduce the average regret given by equation (5) to zero. One such algorithm is Hedge, which maintains a policy based on the sum of regrets, weighted by a learning rate (Freund and Schapire 1997).

$$R_{sum}^T(u, a) = \sum_{t=1}^{T} \eta_t Regret(a, u) \tag{6}$$

The policy update equation of agent a at time t is given by:

$$\pi_t^a = \exp(R_{sum}^T(u, a)) / \sum_u \exp(R_{sum}^T(u, a)) \tag{7}$$

Hedge is no-regret when η_t is chosen carefully, e.g.,$\eta_t \in \Theta(1/\sqrt{t})$ (Nedic and Lee 2014).

New Interpretation and Analysis of COMA

In this section we make three contributions. First, we give a new interpretation of COMA's difference rewards in terms of regret minimization in an identical interest setting. Second, inspired by approaches based on regret minimization in general settings (Hennes et al. 2020), we point out that COMA's update can be slow even in many simpler settings and that a simple modification can lead to improved performance. Third, we show that this connection provides a principled rationale for COMA's use of a bounded softmax.

Justifying COMA's Difference Rewards

To calculate its difference rewards, COMA uses a centralized critic which calculates the counterfactual advantage baseline A^a. A^a compares the value of $Q(s, \mathbf{u})$ and $\sum \pi(u'^a|\tau_a) Q(s, (u'^a, \mathbf{u}^{-a}))$ for agent a and it's action u^a. However, the correctness proof for COMA is independent of the choice of baseline, leaving no theoretical justification for this particular choice other than by analogy to prior successes of difference reward approaches.

The literature on regret-like policy gradient methods (Jin, Keutzer, and Levine 2018; Srinivasan et al. 2018) has emphasized the similarities between advantage calculations and the update rule use by counterfactual regret minimization (Zinkevich et al. 2008).[1] Based on this connection, we provide an interpretation of and justification for COMA's difference reward implementation in terms of regret minimization in cooperative settings.

To make this connection explicit, we focus on a setting with a single state, i.e. an identical interest game. We begin by introducing a variant of COMA for this setting which uses all-actions updates and a tabular representation. We call this version Tabular COMA, and it is given in Algorithm 2.

[1]There is a terminology conflict between these two literatures. The literature on difference rewards views the counterfactual advantage as representing *"How much does the agent currently contributes through the actual action u^a to the overall goal of the system, compared to the counterfactual case when the agent is not present in the system?"* while the literature on regret minimization(Jin, Keutzer, and Levine 2018; Srinivasan et al. 2018) interprets it as *"How much benefit do we do we get by taking a counterfactual action u^a instead of adhering to the actual policy $\pi^a(u^a|\tau_a)$ while keeping the policies of other fixed?"*, resulting in a difference in which part of the update is considered the counterfactual.

Algorithm 2 Tabular COMA update for an agent **a** having **K** actions

```
Result: Update policy for agent $a$ given by $\pi^a$
Sample joint action $\mathbf{u}^{-a}$ from other agents policy $\pi(\mathbf{u}^{-a})$
for k = 1 to K do
    $A^a(u_k) \leftarrow r(u^a_k, \mathbf{u}^a)) - \sum_{u'} \pi^a(u') r(u', \mathbf{u}^{-a})$
    $A^a_{sum}(u_k) \leftarrow A^a_{sum}(u_k) + \eta A^a(u_k)$
    $\pi^a \propto \exp(A^a_{sum})$
end
```

Tabular COMA has four changes from Algorithm 1. First, since we are in a single state setting we dispense with states from the notation. Second, also since we are in a single state setting, rather than use a critic to estimate the expected return $Q(s, \mathbf{u})$ we can directly use the immediate reward when computing the counterfactual advantage baseline. Third, to aid in making the connections to regret minimization explicit we use an all-actions update rule rather than solely updating the action taken as COMA does. Fourth, we assume a softmax policy parameterized by a tabular representation where each parameter gives the logit of the policy for that action and agent. These leads to the given update rule for policies (Hennes et al. 2020, equation (6)).

Tabular COMA is the natural all-actions implementation of COMA in the setting of an indentical interest game. Hennes et al. (2020) derive the same algorithm in a more general (i.e. non-cooperative) setting from Softmax Policy Gradient. As they point out algorithms like Tabular COMA do not quite match up with regret minimization. The issue is the inclusion of the $\pi^a(u_k)$ term in the update for $A^a_{sum}(u_k)$. To exactly match up with the Hedge algorithm it should instead be omitted yielding

$$A^a_{sum}(u_k) \leftarrow A^a_{sum}(u_k) + \eta A^a(u_k) \qquad (8)$$

We refer to Algorithm 2 with the update according to Equation 8 as Tabular COMA-N, with the "N" representing the inclusion of this "NeuRD fix."

With this variants, we can make a precise connection between COMA's difference rewards and regret minimization.

Theorem 1. *In an identical interest game, Tabular COMA-N is equivalent to all agents independently using the Hedge algorithm, in that all agents choose the same policy after each update under both.*

Proof. Immediate from Statement 1 of Hennes et al. (2020) which gives the equivalence of Hedge and NeuRD in the more general setting of normal-form games. □

Theorem 1 provides a satisfying justification for COMA's particular choice of baseline for its difference rewards in terms of an approximation of regret minimization.[2] While regret minimization algorithms like Hedge have a long history and many appealing properties in normal-form games in general, a line of work has shown that they have particularly appealing properties in potential games (also known as congestion games), a generalization of identical interest games. Kleinberg, Piliouras, and Tardos (2009) show that players who use Hedge-like updates end up playing a pure equilibrium for a fraction of time that is arbitrarily close to 1 with probability also arbitrarily close to 1 after a polynomially small transient stage. Mehta, Panageas, and Piliouras (2015) showed that the multiplicative weights algorithm (MW) converges to a pure Nash equilibrium for all but a measure 0 of initial conditions, and hence obtained a stronger guarantee, in identical interest games. Recently, Palaiopanos, Panageas, and Piliouras (2017) showed that a version of the MW update rule converges to equilibrium in potential games. However, if the MW algorithm is run with a constant step-size that is not small enough, the actual sequence of play may exhibit chaotic behavior, even for small 2×2 games. On the other hand, Krichene, Drighès, and Bayen (2015) showed that agents converge to Nash equilibrium in all nonatomic potential games if the same algorithm is run with a decreasing step-size. For bandit setting, Coucheney, Gaujal, and Mertikopoulos (2015) showed that a "penalty-regulated" variant of the MW algorithm converges to ϵ-approximate Nash equilibria in congestion games with bandit feedback. While these results do not directly apply to COMA, they provide a strong intuitive rationale for its performance in cooperative settings.

[2]In addition to the lack of the NeuRD fix, COMA is also missing the necessary weights, in the form of reach probabilities, used by algorithms like CFR (Zinkevich et al. 2008) to ensure correct regret minimization in stateful settings. (Hennes et al. 2020).

COMA vs COMA-N

While our use of Tabular COMA-N rather than Tabular COMA was necessary to make a precise connection to Hedge, Hennes et al. (2020) argue that the NeuRD fix is also important to performance. In particular, if $\pi^a(u_k)$ is small then weighting by it results in slow updates even if u_k is a substantially superior action. They demonstrate that this leads to both slow adaptation or even a lack of convergence in some normal form games. It is not immediate that the same issues arise in cooperative settings. Identical interest games are known to be a particularly well-behaved class of games. In particular, they are special cases of potential games (and even more generally weakly acyclic games) which are relatively forgiving in the sorts of learning dynamics which are guaranteed to converge (Marden, Arslan, and Shamma 2007). We argue that while that while a lack of the NeuRD fix will not prevent convergence, it can substantially harm the rate of convergence. That Tabular COMA is guaranteed to converge in identical interest games, unlike Softmax Policy Gradient in normal form games, is a consequence of the standard analysis of actor-critic algorithms. In their correctness proof for COMA, Foerster et al. (2018) prove that COMA's use of a centralized critic effectively results in an actor-critic update that corresponds to single agent actor-critic updates with a policy parameterized by the parameters of independent actors representing the policy of each individual agent. Thus it is guaranteed to converge to a local optimum of the objective function under standard assumptions for the convergence of actor-critic methods. As the true rewards provide a perfect critic, the same applies to Tabular COMA. Since a local optimum of the shared util-

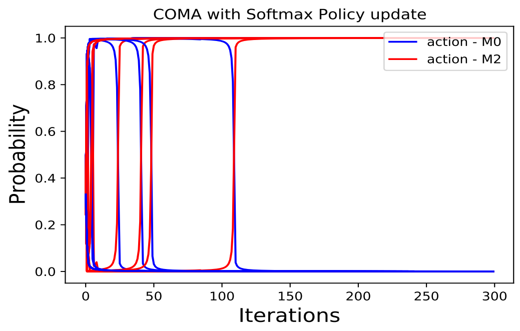

Figure 1: Tabular COMA is slow to update to the unique Nash Equilibrium policy M_2.

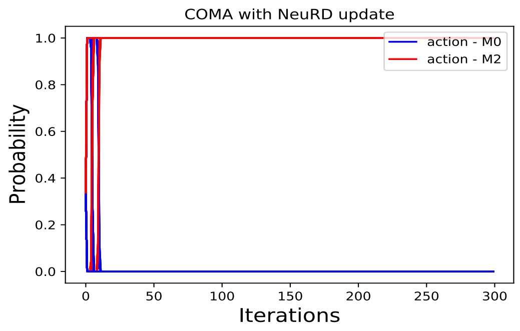

Figure 2: Tabular COMA-N quickly reaches the Nash Equilibrium policy.

ity function is a Nash equilibrium, this shows that failure to include the NeuRD fix does not prevent convergence in identical interest games, unlike in normal form games.

M_0

	L	**R**
U	4	3
D	2	1

M_1

	L	**R**
U	-4	-5
D	-5	-6

M_2

	L	**R**
U	6	-10
D	-20	-20

Table 1: *A 3-player Identical Interest Game*

Consider the three agent identical interest game given in Table 1. In this game Agent 1 can take actions (L, R), Agent 2 can take actions (U, D), and Agent 3 can take actions (M_1, M_2, M_3). This game has a unique Nash Equilibrium (U, L, M_2), but M_2 is a poor choice unless the other two agents are playing their part of the equilibrium. Thus, agent 3 typically needs to "unlearn" playing M_0.

Figure 1 shows the results of 100 runs of Tabular COMA and Tabular COMA-N on this game. Despite the tiny size, with 300 iterations of policy update Tabular COMA couldn't reach the Nash Equilibrium in the time allowed in about half of the runs. This is because, while finding the policy update for agent 3, Tabular COMA use a sample of the policies of agents 1 and 2. This sampling will often lead to bad early rewards for M_2 and thus small values of $\pi^3(M_2)$ and thus slow updates. In contrast, Figure 2 shows that Tabular COMA-N learns rapidly in all runs.

In the experimental section, we show that the benefits of applying the NeuRD fix to COMA hold in richer, stateful settings and not merely in identical interest games.

Justifying Bounded Softmax

The theoretical analysis of COMA by Foerster et al. (2018) is agnostic about the choice of policy representation for each actor. For Tabular COMA we chose a softmax policy to make a precise connection to Hedge possible. The experimental analysis of COMA instead used a bounded softmax policy, where a parameter ϵ is used to put a weight of $1-\epsilon$ on the softmax policy and a weight of ϵ on the uniform policy. We make another connection to prior work showing that this choice leads to strong convergence rate guarantees in identical interest games.

The use of bounded softmax is linked to COMA's updating of actions taken rather than all actions. Thus, rather than Tabular COMA, we use another variant which does not perform all actions updates. In online learning this type of feedback is often referred to as bandit feedback, so we call this variant Bandit COMA-N. It is given in Algorithm 3.

Algorithm 3 Bandit COMA-N update for agent **a**

```
Result: Update policy for agent $a$ given by $\pi^a$
Sample joint action $\mathbf{u}^{-a}$ from other agents policy $\pi(\mathbf{u}^{-a})$
  Sample action $u$ from current policy of an agent $\pi^a$
$A^a(u) \leftarrow r(u, \mathbf{u}^{-a})/\pi^a(u)$
  $A^a_{sum}(u) \leftarrow A^a_{sum}(u) + \eta A^a(u)$
  $\pi^a \propto (1-\epsilon)\exp(A^a_{sum}) + \epsilon/|U|$
```

Since we are working with bandit feedback we do not have access to the term $\sum_{u'} \pi^a(u')r(u', \mathbf{u}^{-a})$ of the Tabular COMA update. However, this term is independent of the choice of u for agent a so simply omitting it has no effect on the policy (this fact in part justifies Theorem 1).

Just as Tabular COMA-N is equivalent to Hedge, Bandit COMA-N is equivalent to ϵ-Hedge with bandit feedback. This algorithm was analyzed by Cohen, Héliou, and Mertikopoulos (2017) in the more general setting of potential games, and the following is a special case of their Theorem 3 restated for our setting.

Theorem 2. *In a generic identical interest game Bandit COMA-N with exploration parameter $\epsilon > 0$ and a suitable choice of learning rate converges almost surely to a $\delta(\epsilon)$ Nash equilibrium, where $\delta(\epsilon) \to 0$ as $\epsilon \to 0$. Furthermore, if the approximate equilibrium Bandit COMA-N converges to puts weight ϵ on the uniform strategy and weight $1-\epsilon$ on a pure strategy for each agent then almost surely this pure strategy profile is a (strict) Nash equilibrium and the convergence to it occurs at a quasi-exponential rate.*

In the theorem, the requirement that the game be generic means that a sufficiently small perturbation of the payoffs does not change the set of Nash equilibria. (A game can be made generic almost surely by an arbitrarily small perturbation to the payoffs, so this assumption is mild.) The quasi exponential rate is much faster than the typical bound of $O(\sqrt{T})$ for the growth of regret. See their paper for the exact bound and requirements on the learning rate.

This analysis provides a principled rationale for the use of bounded softmax by COMA in terms of establishing a convergence rate for this special case. (The theorem does not apply for $\epsilon = 0$ because the variance may grow unbounded, breaking the analysis establishing the convergence rate.) Our experiments show that in some cases using bounded softmax significantly improves the performances over the corresponding regular softmax versions of COMA while in few cases it can lead to slower learning and lower performance because of the unnecessary forced exploration.

Evaluation

In this section, we analyze the importance of the two key features we analyzed theoretically—if the NeuRD fix is used and whether bounded softmax is used—in stateful settings. This gives us four algorithms: COMA, COMA-N (described as Algorithm 4 with the changes highlighted in red), and their variants where the bounded softmax is replaced by a standard softmax Soft-COMA and Soft-COMA-N.

Algorithm 4 COMA-N update for agent **a**

```
Result: Update policy for an agent $a$ given by $\pi^a$ with policy
        network having logits $\upsilon^a$
Given the action of other agents $\mathbf{u}_t^{-a}$, play action $u_t^a$ based
  on agent's history $h_t^a$ for each agent $u_t^a$ and get reward $r_t$
  and next state $s_{t+1}$
  for t = 1 to T do
      Calculate the target network values $y_t$ using the critic
        parameter of the target $\hat{\theta}_t^c$ while batch unrolling states,
        actions and rewards.
end
for t = T to 1 do
    $\Delta\theta^c = \nabla_{\theta^c}(y_t - Q(s_t, \mathbf{u}))^2$
      $\Delta\theta^c \leftarrow \theta^c - \alpha\Delta\theta^c$
      Every C steps reset $\hat{\theta}_t^c = \theta_t^c$
end
for t = 1 to T do
    $A^a(s_t, \mathbf{u}) = Q(s_t, \mathbf{u}) - \sum_u Q(s_t, u, \mathbf{u}^{-a})\pi^a(u|h_t^a)$
      $\Delta\theta^a = \Delta\theta^a + [1/\pi^a(u|h_t^a)]\hat{\nabla}_{\theta^a}\upsilon^a(\theta^a)A^a(s_t, \mathbf{u})$
end
$\theta_{t+1}^a = \theta_t^a + \alpha\Delta\theta^a$
```

We use the implementation of COMA from the repository at https://github.com/QDPP-GitHub/QDPP provided by Yang et al. (2020). For COMA-N, we threshold the range of allowable logits using the same implementation as the OpenSpiel implementation of NeuRD and $\beta = 2$. This thresholding is described by Hennes et al. (2020) as a way to prevent infinite gradients and our testing confirms that performance without it as poor. For all other hyperpameters, we use the values from the repository unless otherwise noted.

We tested all four algorithms on all discrete purely cooperative environments in the repository (some of which are originally due to ma-gym https://github.com/koulanurag/ma-gym). We present results for Switch, Checkers, Blocker, and a Multistep Matrix Game and omit the remaining games since no version of COMA learns meaningfully in them.

Switch Switch is a small grid world navigation game having two or four agents where each agent wants to reach its own home location with either local or shared observations. The challenge is to coordinate with the other agent(s) to navigate through the narrow corridor which can be used only by one agent at a time. The agents need to coordinate to not block the pathway for the other. Although the game provides individual reward to the agents, COMA treats those rewards as global reward and perform our experiments accordingly. A team reward of +5 is given to the agents if one of agent reaches the home cell. The episode ends when all agents have reached their home state or for a maximum of 100 steps in environment. Our experiments in figures 3, 5, 4 and 6 show that using NeurRD fix with COMA-N and Soft-COMA-N never leads to worse performance in this game than COMA and Soft-COMA respectively. Further COMA-N leads to better performance in three out of four versions of the game which shows that this one line fix can lead to adaptive policies without incurring additional costs, consistent with our theoreticak results. The story for softmax updates is more mixed. We observe opposite trends in performance of Soft-COMA vs COMA in figure 5 and 4. Recall that the theoretical advantage of bounded softmax was more stable updates, but this does come at a cost in terms of additional forced exploration due to ϵ. This also explains the performance of Soft-COMA-N vs COMA-N in figure 3.

Checkers This is another grid world setting where the map contains apples and lemons and the two agents have different sensitivity. The first agent scores 5 for the team for an apple and -5 for a lemon. The second, less sensitive agent scores 1 for the team for an apple and -1 for a lemon. Figure 7 shows that all algorithms have essentially the same performance in this setting, with the exception of Soft-COMA-N.[3] This may be because of large fluctuations due to the factor of $1/\pi$ used for bandit settings in policy update equation in 4. However, with a bounded-softmax policy COMA-N manages to perform just like the other variants.

Multistep Matrix Game We use two versions of the Multistep Matrix game introduced by Yang et al. (2020): (v1) is their original version and (v2) is our variant, shown in Figure 8. The only difference is that the -4s in the upper left matrix were +1s in the original version. This environment is unique and designed to be somewhat pathological since it involves many intermediate terminal states (shown in red) and fewer paths leading to later stages. In (v1), Figure 9 shows a perhaps surprising order of performance. This occurs for two reasons. Observe that in this game both players coordinating on either (U,L) or (D,R) yields a score of 10, and all versions

[3]We omit a figure for Checkers-v1 since the results are essentially the same.

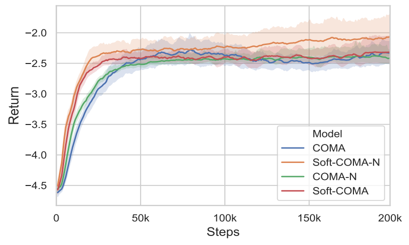

Figure 3: Switch2-v0

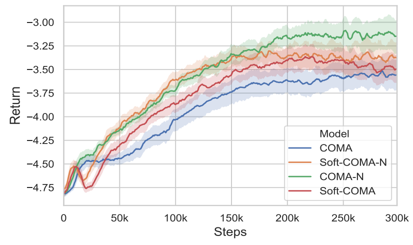

Figure 4: Switch4-v0

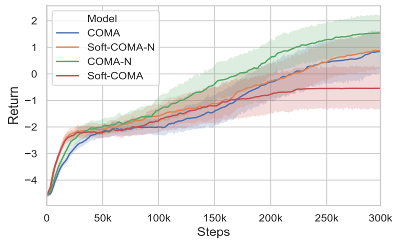

Figure 5: Switch2-v1

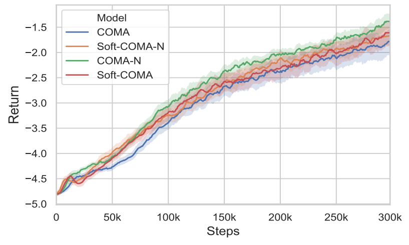

Figure 6: Switch4-v1

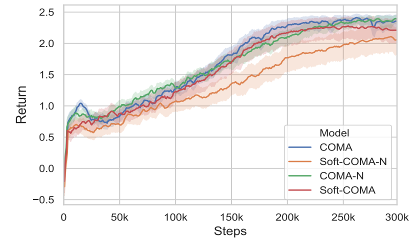

Figure 7: Checkers-v0

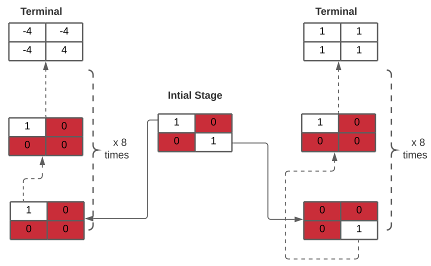

Figure 8: Illustration of Multistep Matrix Game (v2)

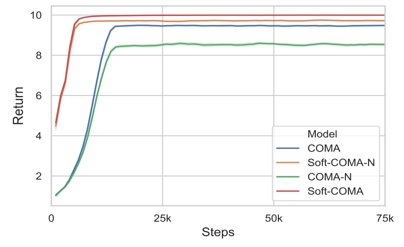

Figure 9: Multistep Matrix Game (v1)

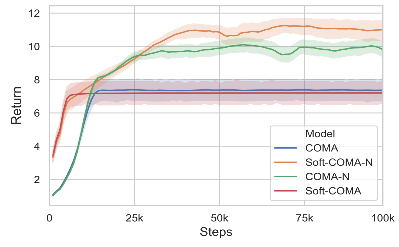

Figure 10: Multistep Matrix Game (v2)

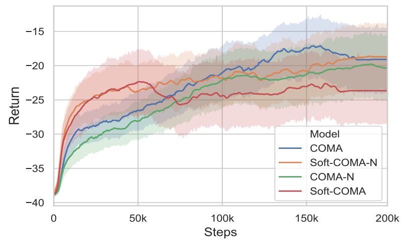

Figure 11: Blocker-v0

essentially converge to this policy. However, the bounded softmax policies anneal their ϵ parameter down to 0.02 over the first 10K steps[4] After that, each agent can still take its intended action at most 99% of the time, so performance degrades due to this forced exploration. Similarly, our clipping of the logits using $\beta = 2$ also puts a limit on the probability assigned to the desired action. These limits entirely explain the performance differences; by way of example, in a sample run after 25K steps Soft-COMA took its desired action in the start state 100% of the time, Soft-COMA-N 99.45%, COMA 98.62%, and COMA-N 97.98%. In (v2), agents who coordinate up (U,L) now have a stronger incentive to learn to play (D,R) in the final stage, which only the more adaptive COMA-N variants do. Again, the lack of forced exploration makes softmax policies slightly better.

Blocker This game requires one of a team of three agents to reach the last row by setting up a situation where moving blockers can only stop two of them. The agents receive -1 reward per time-step before they all reach the destination. The highest reward of the game varies from -6 and -3 depending on the starting points. The agents only have access to decentralized policies and local observations. Figure 11 shows that both versions of softmax initially learn faster, since ϵ in still being annealed hurting the performance of bounded softmax versions. Despite this initial lead, Soft-COMA eventually leads to worst performance but Soft-COMA-N continues to improve along with the bounded softmax variants.

[4]Yang et al. (2020) anneal over a longer period and so report slower learning for COMA than with our tuning.

Conclusion

We provided a new justification for COMA's update rule by connecting it to regret minimization in identical interest games. Based on this we showed that COMA should apply the NRD fix for performance (even if not requried for convergence) and provided a justification for COMA's use of a bounded softmax policy. An important direction for future work is taking this connection beyond the simple setting of identical interest games and analyzing the behavior of regret minimization algorithms in stateful settings.

References

Bernstein, D. S.; Givan, R.; Immerman, N.; and Zilberstein, S. 2002. The complexity of decentralized control of Markov decision processes. Mathematics of operations research 27(4): 819–840.

Cao, Y.; Yu, W.; Ren, W.; and Chen, G. 2012. An overview of recent progress in the study of distributed multi-agent coordination. IEEE Transactions on Industrial informatics 9(1): 427–438.

Cohen, J.; Héliou, A.; and Mertikopoulos, P. 2017. Learning with bandit feedback in potential games. In Proceedings of the 31th International Conference on Neural Information Processing Systems.

Coucheney, P.; Gaujal, B.; and Mertikopoulos, P. 2015. Penalty-regulated dynamics and robust learning procedures in games. Mathematics of Operations Research 40(3): 611–633.

Foerster, J. N.; Farquhar, G.; Afouras, T.; Nardelli, N.; and Whiteson, S. 2018. Counterfactual multi-agent policy gradients. In Thirty-second AAAI conference on artificial intelligence.

Freund, Y.; and Schapire, R. E. 1997. A decision-theoretic generalization of on-line learning and an application to boosting. Journal of computer and system sciences 55(1): 119–139.

Freund, Y.; and Schapire, R. E. 1999. Adaptive game playing using multiplicative weights. Games and Economic Behavior 29(1-2): 79–103.

Hart, S.; and Mas-Colell, A. 2000. A simple adaptive procedure leading to correlated equilibrium. Econometrica 68(5): 1127–1150.

Hennes, D.; Morrill, D.; Omidshafiei, S.; Munos, R.; Perolat, J.; Lanctot, M.; Gruslys, A.; Lespiau, J.-B.; Parmas, P.; Duéñez-Guzmán, E.; et al. 2020. Neural Replicator Dynamics: Multiagent Learning via Hedging Policy Gradients. In Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems, 492–501.

Jianhong Wang, J.; Zhang, Y.; Kim, T.-K.; and Gu, Y. 2020. Shapley Q-value: A Local Reward Approach to Solve Global Reward Games .

Jin, P.; Keutzer, K.; and Levine, S. 2018. Regret minimization for partially observable deep reinforcement learning. In International Conference on Machine Learning, 2342–2351.

Kleinberg, R.; Piliouras, G.; and Tardos, E. 2009. Multiplicative updates outperform generic no-regret learning in congestion games. In Proceedings of the forty-first annual ACM symposium on Theory of computing, 533–542.

Krichene, W.; Drighès, B.; and Bayen, A. M. 2015. Online learning of nash equilibria in congestion games. SIAM Journal on Control and Optimization 53(2): 1056–1081.

Marden, J. R.; Arslan, G.; and Shamma, J. S. 2007. Regret based dynamics: convergence in weakly acyclic games. In Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems, 1–8.

Mehta, R.; Panageas, I.; and Piliouras, G. 2015. Natural selection as an inhibitor of genetic diversity: Multiplicative weights updates algorithm and a conjecture of haploid genetics [working paper abstract]. In Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science, 73–73.

Nedic, A.; and Lee, S. 2014. On stochastic subgradient mirror-descent algorithm with weighted averaging. SIAM Journal on Optimization 24(1): 84–107.

Nguyen, D. T.; Kumar, A.; and Lau, H. C. 2018. Credit assignment for collective multiagent RL with global rewards. In Advances in Neural Information Processing Systems, 8102–8113.

Palaiopanos, G.; Panageas, I.; and Piliouras, G. 2017. Multiplicative weights update with constant step-size in congestion games: Convergence, limit cycles and chaos. In Advances in Neural Information Processing Systems, 5872–5882.

Son, K.; Ahn, S.; Reyes, R. D.; Shin, J.; and Yi, Y. 2020. QOPT: Optimistic Value Function Decentralization for Cooperative Multi-Agent Reinforcement Learning. arXiv preprint arXiv:2006.12010 .

Srinivasan, S.; Lanctot, M.; Zambaldi, V.; Pérolat, J.; Tuyls, K.; Munos, R.; and Bowling, M. 2018. Actor-critic policy optimization in partially observable multiagent environments. In Advances in neural information processing systems, 3422–3435.

Tumer, K.; and Agogino, A. 2007. Distributed agent-based air traffic flow management. In Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems, 1–8.

Wolpert, D. H.; and Tumer, K. 2002. Optimal payoff functions for members of collectives. In Modeling complexity in economic and social systems, 355–369. World Scientific.

Yang, J.; Nakhaei, A.; Isele, D.; Fujimura, K.; and Zha, H. 2018. Cm3: Cooperative multi-goal multi-stage multi-agent reinforcement learning. arXiv preprint arXiv:1809.05188 .

Yang, Y.; Wen, Y.; Chen, L.; Wang, J.; Shao, K.; Mguni, D.; and Zhang, W. 2020. Multi-Agent Determinantal Q-Learning. arXiv preprint arXiv:2006.01482 .

Ye, D.; Zhang, M.; and Yang, Y. 2015. A multi-agent framework for packet routing in wireless sensor networks. sensors 15(5): 10026–10047.

Ying, W.; and Dayong, S. 2005. Multi-agent framework for third party logistics in E-commerce. Expert Systems with Applications 29(2): 431–436.

Yu, R.; Shi, Z.; Wang, X.; Wang, R.; Liu, B.; Hou, X.; Lai, H.; and An, B. 2019. Inducing Cooperation via Team Regret Minimization based Multi-Agent Deep Reinforcement Learning. arXiv preprint arXiv:1911.07712 .

Zhou, M.; Liu, Z.; Sui, P.; Li, Y.; and Chung, Y. Y. 2020. Learning Implicit Credit Assignment for Multi-Agent Actor-Critic. arXiv preprint arXiv:2007.02529 .

Zinkevich, M.; Johanson, M.; Bowling, M.; and Piccione, C. 2008. Regret minimization in games with incomplete information. In Advances in neural information processing systems, 1729–1736.