Fairness in Cooperative Multi-objective Multi-agent Reinforcement Learning using Expected Utility

Farès Chouaki, Aurélie Beynier, Nicolas Maudet, Paolo Viappiani

Fairness in Cooperative Multi-objective Multi-agent Reinforcement Learning using Expected Utility

Farès Chouaki
LIP6, Sorbonne Université
Paris, France
Fares.chouaki@lip6.fr

Aurélie Beynier
LIP6, Sorbonne Université
Paris, France
aurelie.beynier@lip6.fr

Nicolas Maudet
LIP6, Sorbonne Université
Paris, France
nicolas.maudet@lip6.fr

Paolo Viappiani
LAMSADE, PSL
Paris, France
paolo.viappiani@lamsade.dauphine.fr

ABSTRACT

Fairness as equity and compromise across multiple viewpoints is a necessary consideration in any kind of decision that is evaluated from several possibly conflicting perspectives. It is also a property that artificial decision-making agents should uphold to be deployable to real-world problems. However, existing work in sequential decision-making ensures fairness among agents or objectives but struggles with real-world problems that are both multi-agent and multi-objective. Furthermore, research integrating fairness into Multi-Objective Reinforcement Learning (MORL) is focused on optimizing the Scalarized Expected Return (SER) criterion while mostly ignoring the Expected Scalarized Return criterion (ESR). We argue that fairness in MORL should also be investigated under ESR since sometimes it is more suitable when solving problems where fairness matters. In this paper, we study objective-wise fairness in cooperative multi-agent multi-objective decision-making under ESR. We propose the first algorithm that learns efficient decentralized policies while enforcing fairness across objectives under ESR. We identify a key challenge in this setting related to policy conditioning on accumulated returns which hinders decentralized learning and execution, and we present an approach to address it based on inter-agent communication. Experiments on discrete and continuous control tasks demonstrate that our method outperforms existing baselines.

KEYWORDS

Multi-Objective Reinforcement Learning, Multi-agent Reinforcement Learning, Expected Scalarized Return, Fairness

ACM Reference Format:
Farès Chouaki, Aurélie Beynier, Nicolas Maudet, and Paolo Viappiani. 2026. Fairness in Cooperative Multi-objective Multi-agent Reinforcement Learning using Expected Utility. In *Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), Paphos, Cyprus, May 25 – 29, 2026*, IFAAMAS, 13 pages.

Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), C. Amato, L. Dennis, V. Mascardi, J. Thangarajah (eds.), May 25 – 29, 2026, Paphos, Cyprus.

1 INTRODUCTION

Real-world sequential decision-making often requires coordinating multiple agents while balancing conflicting objectives. For instance, in ride-sharing, naive scalar rewards can lead to discriminatory outcomes, such as disadvantaging passengers with reduced mobility, whereas multi-objective rewards enable fairer treatment across passenger types [1].

In many applications, performance must also be reliable at each execution rather than only on average. Consider a group of power plants supplying different city districts: daily coordination must ensure that all districts are adequately served each day, not merely in expectation, to maintain social acceptance. In such settings, fair and efficient policies should satisfy each objective to the same proportion at every execution while maximizing overall performance.

In this paper, we address the problem of computing fair and efficient policies in settings involving multiple agents and multiple objectives. More Specifically, the problems tackled by our approach fall into the team-reward team-utility problems proposed in [23]. Existing reinforcement learning approaches typically simplify the problem, focusing either on a single agent or a single objective. However, such simplifications fail to handle the full complexity of the multi-objective, multi-agent scenario. On one side, single-objective multi-agent algorithms mainly aim to ensure fairness among agents but are not designed to handle multiple objectives. On the other side, single-agent multi-objective methods rarely account for fairness under the Expected Scalarized Return (ESR) criterion, which means they only ensure fairness on average across many policy executions. To date, methods that explicitly address both the multi-agent and multi-objective aspects simultaneously do not yet incorporate fairness considerations.

Contribution: This paper provides an argument as to why ESR can be more suitable for solving multi-objective sequential decision-making problems while ensuring fairness. We benchmark decentralized extensions of MORL algorithms and introduce Dec-EUPG, a first algorithm that solves MOMARL problems under ESR. We point out that this algorithm often rely on unavailable global information even in a decentralized execution context. Thus, we propose Local-Return-Dec-EUPG, which uses local accumulated returns and inter-agent communication to approximate global returns, achieving fair and efficient policies. Evaluations show that Dec-EUPG balances efficiency and fairness per execution, outperforming baselines, with Local-Return-Dec-EUPG achieving results

approaching those obtained by Dec-EUPG without relying on any global information during execution.

2 BACKGROUND AND NOTATIONS

This section introduces the background our approach builds upon and the notations used throughout the remainder of the paper.

2.1 Multi-Objective Decentralized Partially Observable Markov Decision Processes

The Multi-Objective Decentralized Partially Observable Markov Decision Processes (MO-DEC-POMDPs) framework is used to solve multi-objective cooperative multiagent reinforcement learning problems. A MO-DEC-POMDP is a special case of the more general Multi-Objective Partially Observable Stochastic Game (MO-POSG) framework [23], and a multi-objective extension of the DEC-POMDP model [17]. It is defined by the tuple $\langle d, S, \{A^i\}, T, \mu, R, \{O^i\}, \Omega, \gamma \rangle$, where d represents the number of objectives to optimize, S denotes the state space, and A^i is the action space of agent i, with the joint action space $A = \times_i A^i$ (assuming all agents have identical action spaces). The transition function $T : S \times A \times S \to [0, 1]$ maps $(s, \mathbf{a}, s')$ to the probability of transitioning from state $s \in S$ to $s' \in S$ under joint action $\mathbf{a} \in A$, satisfying $\forall s \in S, \mathbf{a} \in A, \sum_{s' \in S} T(s, \mathbf{a}, s') = 1$. The initial state distribution $\mu : S \to [0, 1]$ over S satisfies $\sum_{s \in S} \mu(s) = 1$. The vector-valued reward function $R : S \times A \to \mathbb{R}^d$ maps $(s, \mathbf{a})$ to a reward vector $\mathbf{r} \in \mathbb{R}^d$, reflecting multiple objectives, not multiple agents, and unless stated otherwise, a shared reward structure is assumed where all cooperative agents receive the same reward vector at each timestep. The observation space O^i represents the set of observations agent i may receive (assuming identical observation spaces for all agents), and the observation function $\Omega^i : A \times S \times O^i \to [0, 1]$ maps $(\mathbf{a}, s', o^i)$ to the probability of agent i observing $o^i \in O^i$ upon transitioning to state $s' \in S$ after joint action $\mathbf{a} \in A$, satisfying $\forall \mathbf{a} \in A, s' \in S, \sum_{o^i \in O^i} \Omega^i(\mathbf{a}, s', o^i) = 1$. Finally, $\gamma \in [0, 1)^d$ is a vector of discount factors, one for each objective. If the reward function is agent-specific, the model becomes a MO-POSG [23]. When the model includes only a single agent, it reduces to an MO-POMDP [25]; if this agent has full observability over the state, it corresponds to an MO-MDP [8].

2.2 Fair Aggregation Functions

In multi-objective reinforcement learning, the reward function $R(s, a)$ returns a d-dimensional vector indicating the value of an action in a given state with respect to each of the d objectives. To derive a total ordering over policies most approaches use a scalarization function mapping vectorial returns into scalars. Several scalarization functions aim to balance fairness and efficiency [6]. Following the findings of [13], we use the Nash Social Welfare (NSW) function. NSW is defined as: $NSW(\mathbf{r}) = \prod_{i=1}^{d} \mathbf{r}_i$.

2.3 Optimization Criteria in Multi-Objective Reinforcement Learning

In the multi-objective setting, once a scalarization function is defined, the designer must choose between two optimization criteria [22]:

- **Scalarized Expected Return (SER):** The scalarization function is applied after computing the expected return. The value of a policy π under this criterion is: $V_u^\pi = u\left(\mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t \mid \pi, s_0\right]\right)$.
- **Expected Scalarized Return (ESR):** The scalarization function is applied to the return of each trajectory before taking the expectation: $V_u^\pi = \mathbb{E}\left[u\left(\sum_{t=0}^{\infty} \gamma^t r_t\right) \mid \pi, s_0\right]$.

When the scalarization function is linear, both criteria yield the same optimal policies. However, with nonlinear scalarization, the resulting policies can differ significantly [8, 22, 23]. Section 4.1 discusses these differences and illustrates them using toy examples.

3 RELATED WORK

Our approach addresses Multi-Agent Multi-Objective Reinforcement Learning (MOMARL) using a utility-based framework. This section reviews selected works in single-agent multi-objective RL and single-objective multi-agent RL, with emphasis on methods that explicitly aim for fair and efficient policies.

3.1 Multi-objective Reinforcement Learning

Roijers et al. [21] introduced Expected Utility Policy Gradient (EUPG), which optimizes policies under the ESR criterion with a known non-linear utility function by extending policy gradients to condition on both the state and accumulated returns. Reymond et al. [20] proposed an actor–critic extension that learns a multivariate return distribution, enabling bootstrapping and intra-episode updates. Similarly, Non-Linear Utility MCTS (NLU-MCTS) and Distributional MCTS (D-MCTS) [7] adapt MCTS to compute ESR-optimal policies with known utilities. Several works study fairness in MORL. Cimpean et al. [2] model fairness as additional objectives, while Mandal et al. [13] define axioms for fair scalarization in SER and show that the Nash Social Welfare (NSW) function uniquely satisfies them. Siddique et al. [24] propose GGI-based deep RL methods to enforce fairness under SER in single-agent settings. [15] suggest the use of Lorenz dominance to learn fair policies for problems with many objectives. In contrast, [3] optimizes NSW under ESR. Broader surveys of MORL are provided in [8, 22].

3.2 Multi-agent Reinforcement Learning

We focus on agents learning decentralized policies in a partially observable environment with a shared vectorial reward structure, without a centralized controller. Peshkin et al. [18] extended the REINFORCE algorithm for cooperative single-objective multi-agent settings. Foerster et al. [5] introduced independent-A2C as a baseline for their counterfactual actor-critic algorithm, using value functions and bootstrapping. Fully independent algorithms assume no centralized information, but recent MARL trends show policy-based [5, 12] and value-based [19, 26–28] algorithms leverage centralized information during training for improved policies.

Jiang et al. [10] address fair competitive MARL with Fair Efficient Networks (FEN), a decentralized hierarchical approach using a fair-efficient reward function. Ju et al. [11] tackle the same problem and propose the first algorithm achieving sub-linear regret for α-fairness. [30] introduce a policy gradient method for fairness in cooperative multi-agent systems, similar to our setting, but focused on SER rather than ESR and limited to cases where each user's utility depends on a single agent's policy.

3.3 Multi-agent multi-objective reinforcement learning

In a multi-agent and multi-objective context, Mannion et al. [14] present a theoretical analysis of the difference reward when applied in a multi-objective multi-agent setting and prove that using a difference reward instead of a global reward doesn't alter the relative ordering of rewards. This property allows the usage of the difference reward as a reward-shaping mechanism. [9] introduces the first multipolicy algorithm that can solve cooperative multi-objective multi-agent reinforcement learning problems. They propose a Centralized Training Decentralized Execution approach (CTDE), where each agent, conditioned by the preferences over the objectives, learns to estimate its vectorial state-action value function while a mixing network is used to estimate the global state-action value function. To provide a unified benchmark for both cooperative and competitive MOMARL algorithms, [4] presents MOMALand, the first suite of environments which includes 10 environments that can be used to train and evaluate agents across several MOMARL tasks. Radulescu et al. [23] propose a utility-based taxonomy of multi-agent multi-objective problems. A review of the algorithms and their applications is also presented along with several open research questions.

3.4 Open questions

This section reviewed representative approaches to fairness in multi-objective and multi-agent reinforcement learning. Despite its importance in both MARL and MORL, existing work faces several limitations:

(1) Single-objective MARL solutions like [10] are concerned with fairness between agents, which makes them irrelevant in the common reward setting, while [30] considers fairness under SER and only handles the multi-objective aspect of the problem under specific cases.
(2) MOMARL being a new research field, solutions from that sphere like [9] are general and have not yet tackled fairness issues.
(3) Existing MOMARL solutions have not proposed algorithms for cooperative tasks with known utility functions, and the existing solutions are only applicable to the SER criterion.
(4) MORL solutions that try to learn fair policies regarding the objectives are only applicable for SER and overlook ESR.
(5) Existing MORL algorithms for ESR optimization cannot currently solve multi-agent problems.

In this paper, we address limitations 3, 4, and 5. We argue that fairness should be considered under both SER and ESR criteria. We extend EUPG and MOCAC to the MOMARL setting thus proposing decentralized algorithm for the known utility and common reward setting is proposed. We then point-out a flaw making the proposed algorithms rely on global information that violate the decentralization assumption and suggest a solution based on inter-agent communication. The proposed algorithms are evaluated on a novel MOMARL environment consisting of a multi-agent resource distribution task and on the MO-Multi-Walker Stability problem. Our novel algorithm achieves better performance on the evaluation metrics when compared to the considered baselines.

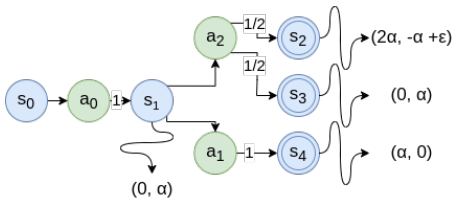


Figure 1: MO-MDP considered in the example

4 FAIRNESS WITH EXPECTED SCALARIZED RETURN

In single-objective RL, a policy's value is its expected return, which in multi-objective RL extends to the Scalarized Expected Return (SER). However, when policy value depends on the utility of individual trajectories, SER-based algorithms can fail, requiring Expected Scalarized Return (ESR) optimization. This section shows how applying SER methods for fairness can produce seemingly unfair policies and how ESR addresses this issue, highlighting the need to distinguish between fairness under SER and ESR optimization.

4.1 ESR vs. SER for ensuring fairness

Example: Consider the MO-MDP shown in Figure 1, with an initial state s_0, three actions a_0, a_1 and a_2, one non-terminal state s_1 and terminal states s_2, s_3 and s_4. The transition function is stochastic with $T(s_0, a_0, s_1) = T(s_1, a_1, s_4) = 1$ and $T(s_1, a_2, s_2) = T(s_1, a_2, s_3) = \frac{1}{2}$. The reward function is a 2-dimensional function of the state given by the curvy arrows going out of a state with $\alpha \in \mathbb{R}^*_+, \epsilon \in \mathbb{R}^*_+$ and $\alpha > \epsilon$.

We compare the deterministic policies π_1 and π_2 defined as $\pi_1(s_0) = \pi_2(s_0) = a_0$, $\pi_1(s_1) = a_1$ and $\pi_2(s_1) = a_2$. Under SER the values of these policies are given by the following:

$$V^{\pi_1}_{NSW} = NSW((\alpha, \alpha)) = \alpha^2$$
$$V^{\pi_2}_{NSW} = NSW\left(\tfrac{1}{2}[(0, 2\alpha) + (2\alpha, \epsilon)]\right) = \alpha\left(\alpha + \tfrac{\epsilon}{2}\right)$$

Under SER, $\forall \epsilon > 0$: $\pi_2 \succ \pi_1$. Thus, under SER the policy π_2 would be selected rather than π_1.

Under ESR, π_1's value remains the same as the one found under SER. However, the value of π_2 changes, and it is given by:

$$V^{\pi_2}_{NSW} = \tfrac{1}{2}[NSW((0, 2\alpha)) + NSW((2\alpha, \epsilon))] = \alpha\epsilon$$

Under ESR, since $\alpha > \epsilon$, we can see that the preference over policies is switched ($\pi_1 \succ \pi_2$).

In both cases, we assert that policy π_1 exhibits superior fairness relatively to the alternative policy π_2. Indeed, π_1 achieves fairer returns during a single execution (since it has a greater value under ESR) and allows for remaining fair on average of multiple executions, meanwhile policy π_2 only achieves fairness when the average of its returns over multiple executions is considered. Indeed, the example highlights two aspects that are undesirable in a fair policy learned by optimizing SER:

• Policies preferred under SER can produce high-variance (and thus unfair) outcomes, depending on the realization. Indeed, the policy preferred by SER achieves stochastic returns which can lead to unfair policies in case the agent finds itself in state s_3 (leading to a return of $(0, 2\alpha)$). Meanwhile, the policy dictated by ESR ensures

that the returns are always as fairly distributed as possible among the objectives.

• Even if it is conditioned by the reward accumulated by the agent until decision time, SER cannot guarantee fairness. Indeed, when the agent finds itself in s_1 with an accumulated reward of $(0, \alpha)$ SER still dictates to perform a_2 over a_1 (since $\pi_2 \succ \pi_1$) even though the fairest action in that situation appears to be a_1.

4.2 ESR vs. SER for ensuring fairness in a multi-agent setting

We now turn to a multi-agent setting and illustrate that considerations related to fairness under ESR and SER still hold. Consider the common reward multi-objective matrix game shown in Table 1. The agents receive the same 2-dimensional reward. We thus only mention this 2-d reward perceived by each agent in each cell of the table. We compare the values of the deterministic joint policy π_1 that always selects the joint action (b, b) and the stochastic policy π_2 that selects uniformly joint actions (a, b) and (b, a). The values

	a	b
a	$(1, 1)$	$(0, 11)$
b	$(11, 0)$	$(5, 5)$

Table 1: MO matrix game considered in the multi-agent example

of these policies under SER are the following:

$$V^{\pi_1}_{NSW} = NSW((5, 5)) = 25$$
$$V^{\pi_2}_{NSW} = NSW\left(\tfrac{1}{2}[(11, 0) + (0, 11)]\right) = 30.25$$

Therefore, under SER $\pi_2 \succ \pi_1$.

If we instead consider the values of these policies under ESR now, only the value of π_2 changes, and we obtain:

$$V^{\pi_2}_{NSW} = \tfrac{1}{2}[NSW((11, 0)) + NSW((0, 11))] = 0$$

Consequently, $\pi_1 \succ \pi_2$, where π_1 ensures that the fairest option is selected at each policy execution.

Notice that even though the matrix game considered is deterministic, the stochastic nature of policy π_2 induces a difference in the value of the policy depending on the optimization criterion considered. We thus argue that, in a multi-objective environment whether stochastic or deterministic, the distinction between ESR and SER is necessary if the agents' policies are stochastic and the common scalarisation function is non-linear (ESR and SER are equivalent when the scalarisation function is linear [8]).

5 FAIR MULTI-OBJECTIVE MULTI-AGENT REINFORCEMENT LEARNING UNDER ESR

The previous section explained the distinction thoroughly between ESR and SER. It presents arguments for why fairness should be studied under both criteria. It illustrates why it can be misleading to learn an objective-wise fair policy under SER for both the single and multi-agent cases. Motivated by these arguments, this section presents an extension of state-of-the-art MORL algorithms under ESR to the multi-agent case. In particular, the proposed algorithm is designed to address problems that fall under the team-reward, team-utility category of the taxonomy proposed in [23]. In the first part of this section, we propose an algorithm that learns decentralized policies for the agents that not only maximize the return on each objective but prioritize objectives which are less satisfied, leading to efficient policies minimizing disparities between objectives. In the latter part, we show that the proposed solution can produce policies that depend on information unavailable at execution time, as they are conditioned on global accumulated vector returns. We therefore introduce an extension of the algorithm applicable when agents have access to both local and aggregated global vector rewards, placing this setting within a sub-scenario of the team-reward team-utility taxonomy in [23]. Such settings naturally arise in real-world applications, including multi-objective vehicle-fleet and wind-farm optimization.

5.1 Decentralized Multi-Objective Reinforcement Learning Algorithms

We propose two decentralized multi-agent algorithms that extend the Expected Utility Policy Gradient (EUPG) [21] and Multi-Objective Categorical Actor-Critic (MOCAC) [20] frameworks to settings where agents learn independently. We opt to extend both these algorithms for the following reasons. On the one hand, EUPG is the natural extension of REINFORCE to the expected utility use-case, it is a fairly simple algorithm to implement and deploy and has a smaller number of learnable parameters compared to other existing algorithms specialized in ESR optimization. MOCAC, on the other hand, was shown to be more sample efficient than EUPG on single agent tasks and through its distributional critic. We argue that these two approaches are complementary in the context of decentralized learning where one favors model simplicity at the cost of sample efficiency (EUPG) and the other ensures more sample efficiency by using more complex models during the training of the policy (MOCAC). The pseudocode of the proposed algorithms is provided in the appendix. The following provides an detailed description of these algorithms.

The first algorithm, Decentralized EUPG (Dec-EUPG), enables each agent to optimize its local policy using scalarized policy gradients based on locally collected trajectories (cf. Appendix C Algorithm 2). At each timestep t, an agent observes its local state and computes the accumulated return before choosing its action. Specifically, we define the past global return as $G_t^- = \sum_{k=0}^{t-1} \gamma^k \mathbf{R}_k$ and the future global return as $G_t^+ = \sum_{k=t}^{T-1} \gamma^k \mathbf{R}_k$,where $\mathbf{R}_k$ represents the global reward received by the agents at timestep k, G_t^- represents the global return accumulated from the beginning of the episode up to t, and G_t^+ is the expected global return from t to the end of the episode. The agent selects an action $a_t^i \sim \pi^i(h_t^i, G_t^-)$, conditioning its policy not only on its local history h_t^i, but also on the accumulated global return so far. After observing the outcome, the agent updates its policy using the scalarized utility of the total return $u(G_t^- + G_t^+)$.

Building on the MOCAC framework, we further propose Decentralized Multi-Objective Categorical Actor-Critic (Dec-MOCAC), an extension of the MOCAC algorithm that incorporates distributional critics in a decentralized setting (cf. Appendix C Algorithm 3). In this algorithm, each agent maintains a distributional value function over multi-objective returns and a policy optimized

with respect to a scalarized expected utility. The value distribution is represented using a discrete set of atoms $\{z_k^{(j)}\}_{k=1}^N$ for each objective j, spaced linearly in a fixed range $[v_{\min}^{(j)}, v_{\max}^{(j)}]$. At each training step, the agent collects local trajectories, projects the target return distribution using the distributional Bellman operator, and updates its critic by minimizing the cross-entropy between predicted and target distributions. The scalarized expected value $Q_t^i = \sum_{k=1}^N u(z_k) \cdot p_{\phi_i}(z_k \mid s_t^i, a_t^i)$ is then used to update the policy via a gradient ascent step.

Together, these algorithms—Dec-EUPG and Dec-MOCAC —offer decentralized and scalable solutions for MOMARL by applying utility-based decision-making and distributional value estimation at the individual agent level. The scalarization function used is the one presented in Section 2.2.

5.2 Lifting the global return conditioning constraint

In multi-objective reinforcement learning (MORL), agents have to condition their policies on the accumulated returns [20, 21]. Extending this to multi-agent settings requires conditioning on the *global return*, the total accumulated rewards across all agents. While accessible during centralized training, global returns are unavailable during decentralized execution, where agents see only local observations. For instance, in a resource-gathering task, an agent may track its own collected resources but remain unaware of others' contributions, making it impossible to compute the global return without additional mechanisms.

Conditioning policies solely on local returns or removing conditioning altogether can alleviate the need for global information but often falls short in achieving optimal policies. Local return conditioning provides agents with incomplete utility signals, which may misalign individual behaviors and hinder effective coordination, leading to suboptimal collective outcomes. Similarly, policies without any return conditioning are proven to be suboptimal in the single agent-case [21].

To address this, we position ourselves in a context where multi-agent communication is allowed during both training and execution of the policy and communication is limited to the observation range of each agent. We assume that each agent is characterized by an observation radius and an agent can only communicate with the agents that are within its radius. We propose Local-Return-Dist-EUPG (Algorithm 1). This algorithm is similar to Dec-EUPG however, during training, agents not only have access to the global reward which is used as reward signal (R_t) to optimize the agents' policies, they also can keep track of their own vectorial accumulated return($\mathbf{acc\_r^i}$). Agents also track the vectorial accumulated returns of their teammates ($\mathcal{B}^i$). At decision time, each agent conditions its policy only by its observation history and a proxy of the global accumulated return ($\mathbf{Global\_acc\_R_t^i}$). This proxy is build by each agent through the combination of its local accumulated returns and its beliefs on the accumulated returns of its teammates. After each step, each agent updates its beliefs about the accumulated returns of its neighbors and updates its own accumulated return tracker. At the end of the episode each agent updates its policy using the trajectory they gathered during that episode. Tracking other agents accumulated return is crucial to ensure decentralized execution making inter-agent communication a necessity for the algorithm. This communication facilitates an approximation of the global return, enabling decentralized policies to incorporate near-global conditioning without relying on a central entity. Moreover, we conjecture that in MO-Dec-POMDPs where the reward function is stochastic multi-agent communication is necessary in order to learn the optimal policy. In our solution, agents communicate the total return they accumulated whenever communication is possible (whenever two agents are in range of communication they

Algorithm 1: Local-Return-Dec EUPG with Communication

Data:
$tr\_timesteps > 0$ // Number of training timesteps
$u : \mathbb{R}^d \to \mathbb{R}$ // Scalarisation function
$\gamma \in (0, 1]$ // Discount factor

Initialize policy parameters and hidden states for each agent;
$steps \leftarrow 0$;
while $steps < tr\_timesteps$ **do**
  Reset environment and observe initial state S_0;
  for *each agent i* **do**
    accumulated returns $\mathbf{acc\_r}^i \leftarrow \mathbf{0}_d$;
    neighbor local return $\mathcal{B}^i \leftarrow \{\mathbf{0}_d\}_{i=1}^{N_{\text{agents}}-1}$;
    trajectory $\mathcal{T}^i \leftarrow []$;
  end
  while *episode not done and* $steps < tr\_timesteps$ **do**
    for *each agent i* **do**
      Update global accumulated return proxy $\mathbf{Global\_acc\_R_t^i} \leftarrow acc\_r^i + \sum_{j \in B^i} j$;
      Select action a_t using policy, considering state action history and $\mathbf{Global\_acc\_R_t^i}$;
    end
    Execute joint action A_t in environment, observe local rewards $\mathbf{r}_t$ and global reward $\mathbf{R}_t$ and next state S_{t+1};
    for *each agent i* **do**
      Communicate local return to each visible agent;
      Update neighbors' accumulated return with communicated rewards;
      Update own accumulated return with $\mathbf{r}_t^i$;
      Store transition $(S_t^i, Global\_acc\_R_t^i, A_t^i, R_t, S_{t+1}^i)$ in $\mathcal{T}^i$;
    end
    Increment *steps*;
  end
  for *each agent i* **do**
    Update policy parameters using trajectory scalarization function u;
    // the global accumulated return proxy ($Global\_acc\_R^i$) is used to condition the policy while the global R_t is used as reward signal
  end
end

exchange messages)[1], minimzing the number of messages sent and learning better communication policies is left as future work. Continuing on the resource-gathering example, during training, agents condition their policy on the amount of resources they collected and the shared information from other agents about their collected resources, thereby constructing a more accurate estimate of the global return to guide their decision-making. This approach allows agents to learn a policy that not only favors fair and efficient collection of objectives but also leverages communication to enhance coordination and optimize collective outcomes.

6 EXPERIMENTAL RESULTS

This section presents the evaluation scenarios that the algorithms are evaluated on, the baselines considered for comparison with our algorithm, the evaluation metrics used to assess the performances of the agents, and the results obtained. This section concludes with a thorough discussion of the results.

6.1 Evaluation scenarios

We evaluate our approach on a partially observable delivery task where multi-agent coordination is needed to achieve optimal scores on the evaluation criteria. These environments will be referred to as RG:dO-nA where d represents the number of objectives and n the number of agents present in the environment.

The first evaluation settings are simple ones where the household positions are fixed on a 10*10 grid world. Two tasks are considered: a 2-agent bi-objective task and 3-agent tri-objective task. In the first task, each agent can deliver resources to two types of households, the number of households of each type present in the environment is 3, whereas each agent carries enough resources to accommodate up to 2 houses. In the second task, each agent can deliver resources to 3 households.[2] The environment is described in Appendix A.

We use this setting to validate our approach and prove that decentralized policies can indeed be learned by the agents using our algorithms. Hence, weshowcase the superiority of the proposed algorithms compared to existing Single Objective MARL baselines.

To evaluate the scaling abilities of our approach, we propose a more complex setting where the position of the households change after each episode. We consider two scenarios under this setting. The first one is similar to the first scenario of the previous setting where two agents have to fairly distribute the resources at their disposal on two types of households, however, each agent now can accommodate up to 3 households among the 6 households of each type that are in the environment. The second scenario involve 4 agents and 4 objectives, each agent can only accommodate two households on grid of size 20*20.

Moreover, to evaluate the abilities of Local-Return-EUPG, we propose a 2-agents 3-objectives resource gathering task. In this task we consider a 10*10 grid world environment with a stochastic reward function. A more detailed description of the environment is given in appendix A.

[1]A simple way of reducing the amount of messages sent between agents would be for each agent to keep track of what they sent to each teamate and only send a message when that information becomes obsolete

[2]The code used to generate the results in available at https://gitlab.com/chouakifares/fairness-in-cooperative-multi-objective-multi-agent-reinforcement-learning-using-expected-utility

Finally, we also show how our proposed algorithm can be used to solve complex continuous control tasks using the MO-Multi-Walker Stability Task. MO-Multi-Walker Stability (referred to as MO-MWS in Section 6.3) has been proposed in MOMALand [4][3]. In this problem agents have to coordinate their movements to jointly carry a package placed on top of their heads while balancing it. At each step, the agents receive a common global bi-objective reward consisting of the distance traveled towards the end of the level during that step and a penalty based on the change of angle in the package. Thus, learning a fair policy in this case leads the agents to maximize the distance traveled while minimizing eventual damage that the package can incur during transportation.

6.2 Baselines

Since fairness in MOMARL algorithms under the ESR criterion is unexplored, no existing algorithm effectively addresses the evaluation scenarios. MORL algorithms like MOMIX [9] and multi-agent extensions [24] cannot ensure fairness over a single execution. Similarly, fairness-focused MARL algorithms prioritize fairness across multiple executions and struggle with MOMARL's multi-objective nature. For example, [30] is unable to handle cases where the value of an objective depends on the policy of several agents, and FEN [10] cannot handle multiple objectives or cooperative settings. For this reason, we propose to compare our solution to a centralized single-agent solution and adaptation of single-objective decentralized RL algorithms to the multi-objective case like [20, 21]. We also adapt the approach proposed in [10] to the task of ensuring fairness across objectives in two ways.

Centralized single-agent baseline: This baseline assumes that there exists a centralized agent controlling all the agents at once. At each step of an episode, the controller receives the joint observation of the agents, the global vectorial reward received by the agents, and performs a joint action in the environment. This baseline can alleviate problems related to the partial observability of the environment and multi-agent credit assignment. However, it may suffer from exploration issues due to a larger action space.

Single-objective decentralized baseline: This baseline applies the A2C [16] algorithm on each agent independently. To ensure that the learned policy optimizes the expected utility, the reward function of the MO-decPOMDP is transformed such that as long as the episode is not over, the agent receives at each step t a reward of $r_t = 0$ while the actual reward vector $\mathbf{r}_t$ is accumulated in a backup variable $\mathbf{R} = \sum_{t=0}^{T} \gamma^t r_t$. At the end of an episode all the agents receive the scalarized accumulated reward $r_T = NSW(\mathbf{R})$.

FEN-agents baseline: In this approach, each agent is assigned a single objective to optimize throughout the episode, following the Fair-Efficient Network (FEN) approach as described in [10]. Each agent learns a policy tailored to maximize its designated objective, leveraging the FEN framework to ensure fairness across agents while optimizing their respective objectives.

FEN-objectives baseline: This approach extends the FEN framework by training each agent to learn $n_{\text{objectives}}$ policies, where each

[3]Among all the proposed environments in MOMALand, only MO-MWS was tested because most of them do not model cooperative MOMARL tasks while the rest are very similar to the resource distribution and resource gathering tasks considered in the rest of the experiments

policy maximizes one of the objectives. The agent-specific meta-policy is then employed to select which objective an agent should prioritize at each step or episode. This meta-policy dynamically chooses the objective to optimize based on the current state and accumulated global return, aiming to balance fairness and efficiency across all objectives.

To ensure a fair comparison between our approach and these baselines, we use the EUPG and MOCAC algorithms for the centralized baseline and the policy gradient, the A2C algorithm for the single-objective RL baseline and PPO for the FEN baselines.

6.3 Results

This section compares the results[4] obtained by each approach considered on both evaluation tasks presented in Section 6.1. Essentially this section aims to answer the following questions:

- Can current state-of-the-art MORL algorithms made for ESR optimization be used to learn decentralized policies?
- Are the policies learned by the proposed algorithms able to ensure fairness between the objectives?
- Are MORL algorithms more interesting than their Single Objective RL counter-parts when it comes to solving MOMARL tasks?
- During the execution stage of the learned policies, is it possible to replace the global accumulated reward used to condition the policy by a proxy obtained through inter-agent communication?

Appendix D explores the scaling abilities of our approach in terms of agents, objectives and state-space.

Can current state-of-the-art MORL algorithms made for ESR optimization be used to learn decentralized policies?
Figure 2 compares the results obtained by the decentralized algorithms and the centralized baselines on the evaluation tasks. Figures 2a and 2b show that the centralized variant of MOCAC is the fastest learning algorithm able to achieve efficient policies after 2.5 million steps in the environment. The centralized and decentralized EUPG algorithms can achieve comparable performance on the task with 2 agents but require more interactions with the environment. Surprisingly the decentralized variant of the MOCAC algorithm is not able to learn a policy better than its initial policy on the task considered. The continuous control evaluation task shows the limitation of centralized controllers which fail to solve the task, likely due to the increased size of the action space for these algorithms exacerbated by its continuous nature. The decentralized MOCAC algorithm achieves only slightly better returns than these baselines. Nonetheless, the decentralized EUPG algorithm is able to solve the task and achieve better performances than the centralized baselines.

Are the policies learned by the proposed algorithms able to ensure fairness between the objectives?
The results presented in Figures 2 illustrate the evolution of the scalarized global returns achieved by each algorithm. In the resource distribution task with two agents and two objectives, decentralized EUPG learns an efficient and fair policy, achieving a mean discounted return close to 4, a performance that is only attainable when agents coordinate to allocate resources to households of distinct types.

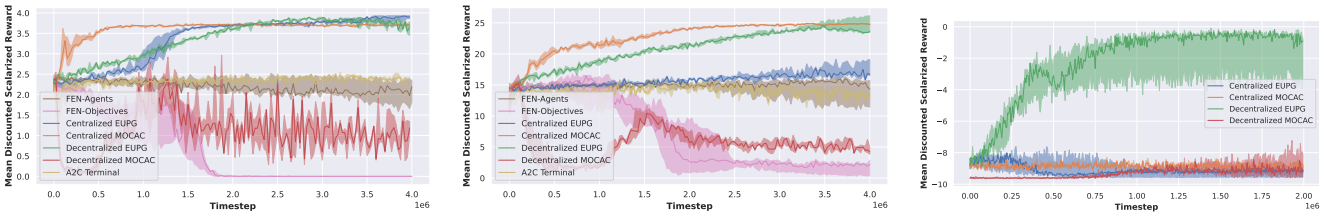
This observation extends to the more challenging three-agent, three-objective setting. In the RD:3O–3A scenario, each agent can accommodate three households, and efficiency requires serving all nine available slots. Among these efficient policies, objective-wise unbalanced allocations are provably suboptimal with regards to the NSW scalarization function. Indeed, policies prioritizing one or two household types yield return vectors such as $(4, 4, 1)$ or $(4, 3, 2)$, whose scalarized returns are upper bounded by 16 and 24, respectively. In contrast, only the objective-wise fair allocation $(3, 3, 3)$ achieves the maximal scalarized return of 27. Consequently, achieving scalarized returns above 24 necessarily implies fairness across objectives. Since Dec-EUPG reaches an average scalarized (discounted) return of 25, the learned policies must correspond to fair allocations.

Finally, Dec-EUPG also scales to more complex MOMARL tasks with continuous action spaces (Figure 2c) and larger numbers of agents and objectives (Figure 5).

Are MORL algorithms more efficient than their SORL counter-parts when it comes to solving MOMARL tasks?
Figures 2a and 2b also give the performances of the single-objective decentralized baselines. These results show that like the decentralized MOCAC algorithm, applying the A2C algorithm in a decentralized manner on the transformed Dec-POMDP does not yield interesting policies proving the superiority of the decentralized EUPG algorithm over decentralized single-objective RL algorithms. Moreover, the adaptations of the FEN approach to our use-case do not yield efficient policies , which can be explained by the difference in optimization criterion considered by this approach. Finally, we stress that our approach offers lower complexity and greater ease of deployment in real-world applications.

During the execution stage of the learned policies, is it possible to replace the global accumulated reward used to condition the policy by a proxy obtained through inter-agent communication?
This scenario is important to consider as in real-life problems it is easier for an agent to keep track of the reward it has accumulated than the reward accumulated by all the agents. However agents lacking knowledge of the reward accumulated by their teammates may fail to properly coordinate, leading to suboptimal collective behavior. Figure 3 compares the performances obtained by i) the decentralized EUPG conditioned on the global accumulated return of all agents, ii) a variant where agents' policies are not conditioned by the returns at all, iii) and the variant where agents are trained using Local-Return-Dec-EUPG conditioned only on the own accumulated return and their beliefs on the accumulated return of their teammates. These experiments were possible only on the resource distribution environment where it is possible to define such local rewards. We observe that when using Local-Return-EUPG, performances are somewhat hindered compared to when agents condition their policy with the true value of accumulated returns but it still outperforms the variant where agents do not condition their policy by the accumulated return. This difference in performance is explained by the nature of the task (cf Appendix A). The agent that collects the ordinary chest needs to know which makes conditioning on the global accumulated return important.However,

[4]The results have been obtained using the NSW scalarisation but they remain valid for the Ordered Weighted Average (OWA) [29] family of functions as long as the weights are decreasing.



(a) RD:2O-2A (b) RD:3O-3A (c) MO-MWS

Figure 2: Comparison of the proposed algorithms to the baselines

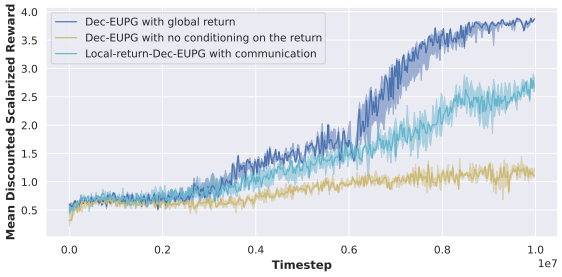


Figure 3: Comparison of the Local-Return-EUPG algorithm to Dec-EUPG conditioned on the accumulated global return and DEC-EUPG with no return conditionning

this violates the decentralization assumption. Thus Local-Return-Dec-EUPG constitutes a bridge between these two approaches and provides a first solution to train and deploy decentralized agents that solve MOMARL tasks efficiently while ensuring fairness at each execution of the policy.

6.4 Discussion

This section analyzes and discusses the results obtained along with the limitations of the proposed algorithms.

Decentralized MOCAC is not suited for decentralized learning: From the experimental results, we notice that the centralized MOCAC algorithm works well. Although it suffers from early convergence, it is able to learn efficient policies the fastest. On the contrary, the decentralized MOCAC algorithm with local or global reward conditioning performs the worse and is unable to learn efficient policies. Given that the decentralized EUPG algorithm and the centralized MOCAC algorithm both achieve good performances we argue that the reason for the poor performances of the Decentralized MOCAC algorithm is that the agent specific critics are unable to learn correct state-value distributions since they only observe the agent's state and action whereas the state's value is determined by the joint action of all the agents.

Decentralized EUPG a first solution for achieving fairness in MOMARL under ESR: The results presented in Section 6.3 show that the algorithms we propose can solve cooperative multi-agent multi-objective sequential decision-making problems, with both discrete and continuous action spaces, while ensuring fairness towards the objectives. The local-reward variant can be as efficient as the global-reward variant when such a transformation is possible. This makes our algorithms more practical and suitable for deployment in real-world applications. We argue that the proposed solutions are most suited for applications where fairness needs to be guaranteed at each execution such as ethical multi-objective reinforcement learning.

Limitations of the Decentralized EUPG algorithm: The Decentralized EUPG algorithm can handle scenarios involving a limited number of agents and objectives. Indeed, as an on-policy algorithm, it struggles with sample efficiency.

The current version of the algorithm could be further enhanced by centralized training. Moreover, the optimization of the communication strategy used in the current implementation of Local-Return-EUPG remains an open direction for future work. Finally, it would be interesting to study whether a centralized training with decentralized execution approach would improve the performances of MOCAC.

7 CONCLUSION

This paper identifies a core limitation in current fair multi-objective reinforcement learning algorithms: while they enforce fairness across multiple executions of a policy, they fail to ensure fairness among the objectives over a single execution. To address this issue, we proposed two novel decentralized MOMARL algorithms. These algorithms extend existing state-of-the-art MORL algorithms for ESR [20, 21]. Experiments on multi-objective multi-agent discrete and continuous control tasks showed that the decentralized EUPG algorithm is a first solution to problems where fairness across objectives is important. We also show that the decentralized MOCAC algorithm is not able to solve the evaluation tasks which motivates future work on eventual Centralized Training with Decentralized Execution (CTDE) algorithms for multi-objective problems under ESR. Finally, experimental results on the resource distribution task shows that leveraging local rewards for training and conditioning agents's policies yields performance comparable to that achieved with a global reward making our solution directly applicable to problems where such local rewards can be defined. As future work, we plan on extending CTDE methods for the multi-objective case and on potentially using these technics to propose outer-loop multi-policy algorithms for MOMARL tasks under ESR.

REFERENCES

[1] Tarek Chouaki and Sebastian Hörl. 2024. Comparative assessment of fairness in on-demand fleet management algorithms. In *The 12th Symposium of the European Association for Research in Transportation (hEART)*. Espoo, Finland. https://hal.science/hal-04551002

[2] Ioana Alexandra Cimpean, Catholijn M Jonker, Pieter Libin, and Ann Nowe. 2023. A Multi-objective Framework For Fair Reinforcement Learning. 1–11. https://modem2023.vub.ac.be Multi-Objective Decision Making Workshop 2023, MODeM 2023 ; Conference date: 01-10-2023 Through 01-10-2023.

[3] Ziming Fan, Nianli Peng, Muhang Tian, and Brandon Fain. 2023. Welfare and Fairness in Multi-objective Reinforcement Learning. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems* (London, United Kingdom) *(AAMAS '23)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 1991–1999.

[4] Florian Felten, Umut Ucak, Hicham Azmani, Gao Peng, Willem Röpke, Hendrik Baier, Patrick Mannion, Diederik M. Roijers, Jordan K. Terry, El-Ghazali Talbi, Grégoire Danoy, Ann Nowé, and Roxana Rădulescu. 2024. MOMAland: A Set of Benchmarks for Multi-Objective Multi-Agent Reinforcement Learning. arXiv:2407.16312 [cs.MA] https://arxiv.org/abs/2407.16312

[5] Jakob N. Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. 2018. Counterfactual multi-agent policy gradients. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence* (New Orleans, Louisiana, USA) *(AAAI'18/IAAI'18/EAAI'18)*. AAAI Press, Article 363, 9 pages.

[6] Christophe Gonzales and Patrice Perny. 2020. Multicriteria Decision Making. In *A Guided Tour of Artificial Intelligence Research*, Pierre Marquis, Odile Papini, and Henri Prade (Eds.). Knowledge Representation, Reasoning and Learning, Vol. I. Springer, 519–548. https://doi.org/10.1007/978-3-030-06164-7_16

[7] Conor F. Hayes, Mathieu Reymond, Diederik M. Roijers, Enda Howley, and Patrick Mannion. 2023. Monte Carlo tree search algorithms for risk-aware and multi-objective reinforcement learning. *Autonomous Agents and Multi-Agent Systems* 37, 2 (April 2023), 37. https://doi.org/10.1007/s10458-022-09596-0

[8] Conor F. Hayes, Roxana Rădulescu, Eugenio Bargiacchi, Johan Källström, Matthew Macfarlane, Mathieu Reymond, Timothy Verstraeten, Luisa M. Zintgraf, Richard Dazeley, Fredrik Heintz, Enda Howley, Athirai A. Irissappane, Patrick Mannion, Ann Nowé, Gabriel Ramos, Marcello Restelli, Peter Vamplew, and Diederik M. Roijers. 2022. A practical guide to multi-objective reinforcement learning and planning. *Autonomous Agents and Multi-Agent Systems* 36, 1 (April 2022). https://doi.org/10.1007/s10458-022-09552-y

[9] Tianmeng Hu, Biao Luo, Chunhua Yang, and Tingwen Huang. 2023. MO-MIX: Multi-Objective Multi-Agent Cooperative Decision-Making With Deep Reinforcement Learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45, 10 (2023), 12098–12112. https://doi.org/10.1109/TPAMI.2023.3283537

[10] Jiechuan Jiang and Zongqing Lu. 2019. *Learning fairness in multi-agent systems*. Curran Associates Inc., Red Hook, NY, USA.

[11] Peizhong Ju, Arnob Ghosh, and Ness Shroff. 2024. Achieving Fairness in Multi-Agent MDP Using Reinforcement Learning. In *The Twelfth International Conference on Learning Representations*. https://openreview.net/forum?id=yoVq2BGQdP

[12] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. 2017. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Long Beach, California, USA) *(NIPS'17)*. Curran Associates Inc., Red Hook, NY, USA, 6382–6393.

[13] Debmalya Mandal and Jiarui Gan. 2022. Socially fair reinforcement learning. *arXiv preprint arXiv:2208.12584* (2022).

[14] Patrick Mannion, Sam Devlin, Jim Duggan, and Enda Howley. 2018. Reward shaping for knowledge-based multi-objective multi-agent reinforcement learning. *The Knowledge Engineering Review* 33 (2018), e23. https://doi.org/10.1017/S0269888918000292

[15] Dimitris Michailidis, Willem Röpke, Diederik M. Roijers, Sennay Ghebreab, and Fernando P. Santos. 2024. Scalable Multi-Objective Reinforcement Learning with Fairness Guarantees using Lorenz Dominance. arXiv:2411.18195 [cs.LG] https://arxiv.org/abs/2411.18195

[16] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous Methods for Deep Reinforcement Learning. In *Proceedings of The 33rd International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 48)*, Maria Florina Balcan and Kilian Q. Weinberger (Eds.). PMLR, New York, New York, USA, 1928–1937. https://proceedings.mlr.press/v48/mniha16.html

[17] Frans A Oliehoek, Christopher Amato, et al. 2016. *A concise introduction to decentralized POMDPs*. Vol. 1. Springer.

[18] Leonid Peshkin, Kee-Eung Kim, Nicolas Meuleau, and Leslie Pack Kaelbling. 2000. Learning to cooperate via policy search. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence* (Stanford, California) *(UAI'00)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 489–496.

[19] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder De Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. 2020. Monotonic value function factorisation for deep multi-agent reinforcement learning. *J. Mach. Learn. Res.* 21, 1, Article 178 (Jan. 2020), 51 pages.

[20] Mathieu Reymond, Conor Hayes, Denis Steckelmacher, Diederik Roijers, and Ann Nowe. 2023. Actor-critic multi-objective reinforcement learning for non-linear utility functions. *Autonomous Agents and Multi-Agent Systems* 37 (04 2023). https://doi.org/10.1007/s10458-023-09604-x

[21] Diederik M. Roijers, Denis Steckelmacher, and Ann Nowé. 2020. Multi-objective reinforcement learning for the expected utility of the return. 2018 Adaptive Learning Agents, ALA 2018 - Co-located Workshop at the Federated AI Meeting, FAIM 2018 ; Conference date: 14-07-2018 Through 15-07-2018.

[22] D. M. Roijers, P. Vamplew, S. Whiteson, and R. Dazeley. 2013. A Survey of Multi-Objective Sequential Decision-Making. *Journal of Artificial Intelligence Research* 48 (Oct. 2013), 67–113. https://doi.org/10.1613/jair.3987

[23] Roxana Rădulescu, Patrick Mannion, Diederik M. Roijers, and Ann Nowé. 2019. Multi-objective multi-agent decision making: a utility-based analysis and survey. *Autonomous Agents and Multi-Agent Systems* 34, 1 (Dec. 2019). https://doi.org/10.1007/s10458-019-09433-x

[24] Umer Siddique, Paul Weng, and Matthieu Zimmer. 2020. Learning fair policies in multiobjective (deep) reinforcement learning with average and discounted rewards. In *Proceedings of the 37th International Conference on Machine Learning (ICML'20)*. JMLR.org, Article 826, 11 pages.

[25] Harold Soh and Yiannis Demiris. 2011. Evolving policies for multi-reward partially observable markov decision processes (MR-POMDPs). In *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation* (Dublin, Ireland) *(GECCO '11)*. Association for Computing Machinery, New York, NY, USA, 713–720. https://doi.org/10.1145/2001576.2001674

[26] Kyunghwan Son, Daewoo Kim, Wan Ju Kang, David Earl Hostallero, and Yung Yi. 2019. Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In *International conference on machine learning*. PMLR, 5887–5896.

[27] Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z. Leibo, Karl Tuyls, and Thore Graepel. 2018. Value-Decomposition Networks For Cooperative Multi-Agent Learning Based On Team Reward. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems* (Stockholm, Sweden) *(AAMAS '18)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 2085–2087.

[28] Jianhao Wang, Zhizhou Ren, Terry Liu, Yang Yu, and Chongjie Zhang. 2021. QPLEX: Duplex Dueling Multi-Agent Q-Learning. arXiv:2008.01062 [cs.LG] https://arxiv.org/abs/2008.01062

[29] R.R. Yager. 1988. On ordered weighted averaging aggregation operators in multicriteria decisionmaking. *IEEE Transactions on Systems, Man, and Cybernetics* 18, 1 (1988), 183–190. https://doi.org/10.1109/21.87068

[30] Matthieu Zimmer, Claire Glanois, Umer Siddique, and Paul Weng. 2021. Learning Fair Policies in Decentralized Cooperative Multi-Agent Reinforcement Learning. In *Proceedings of the 38th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 139)*, Marina Meila and Tong Zhang (Eds.). PMLR, 12967–12978. https://proceedings.mlr.press/v139/zimmer21a.html

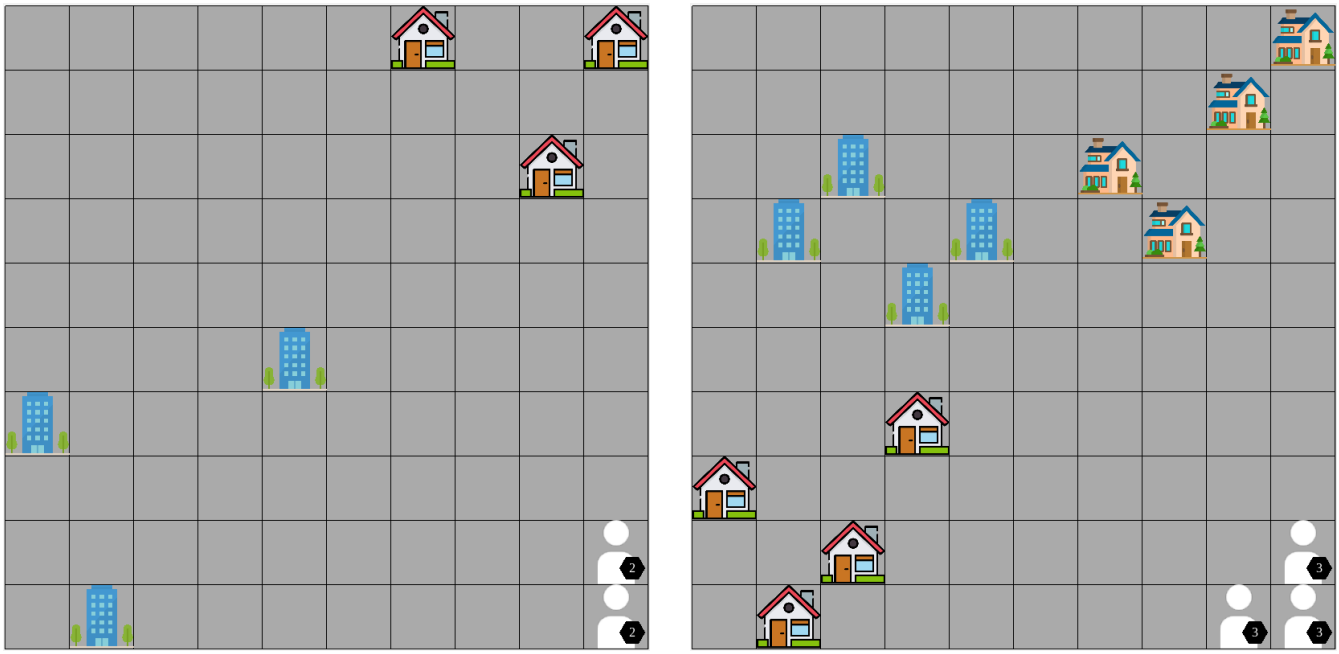
Fairness in Cooperative Multi-objective Multi-agent Reinforcement Learning using Expected Utility – Supplementary Material

A ENVIRONMENT DESCRIPTION

In this section, we describe the environments used to evaluate our agents.

A.1 Resource Distribution

This environment was inspired by the Pressureplate [5] environment and models cooperative multiobjective multiagent sequential decision-making tasks. The agents have to solve the task of resource distribution to different types of households. The environment is modeled as a 2d grid world of custom size and is set to 10 × 10 by default. At each timestep, the agents can move north, south, east, and west on the grid or stay still. A moving action is only successful if the cell the agent is trying to reach does not contain another agent on it. At each new episode, agents and households are replaced at their initial positions as shown in Figure 4.The goal of the agents is to distribute resources among households fairly. An agent can serve a resource to a household by moving toward the cell where the household is located. An agent's observations consist of its relative position in the grid, the total amount of resources it still has not served, and the amount of resources it served for each type of household. Agents also use sensors to observe other agents and all the households that haven't been accommodated yet. The sensor radius can be customized and is set to 2 by default. Under the global reward setting, after each timestep, each agent receives a vector reward of size d (with d representing the number of objectives in the environment), where the i-th component of that vector is the number of households of type i that all the agents accommodated during that timestep. Under the local reward setting the difference is that the i-th component of the vector represents the number of households of type i that were accommodated only by the agent during that timestep. Note that the reward is normalized by the maximum reward achievable by the agent to stabilize training.



(a) Task with 2 agents and 2 objectives

(b) Task with 3 agents and 3 objectives

Figure 4: Example of the resource distribution task the agents learn to solve

Fixed households scenarios: In these scenarios, the number of household of each type present in the environment is 4. In the bi-objective case, each agent can serve at most 2 households per episode. In the scenario where three types of households exist in the environment, each agent can accommodate up to 3 households. The episode ends when all agents exhaust all the resources they had available or after 500 timesteps of the environment.

[5] https://github.com/uoe-agents/pressureplate

Randomized households scenarios: In these scenarios, the number of household of each type present in the environment is 6. In the bi-objective case, each agent can serve at most 3 households per episode. In the scenario where four types of households exist in the environment, each agent can accommodate up to 2 households. The episode ends when all agents exhaust all the resources they had available or after 200 timesteps of the environment.

A.2 Resource Gathering

In this task we consider a 10*10 grid world environment containing four chests, three ordinary chest and one enchanted chest. One agent (a_1) can only collect one ordinary chest while the other (a_2) can collect two ordinary chest or the enchanted chest. Each ordinary chest contains exactly one resource of each objective, while the enchanted chest contains two resources of two objectives. The agents can observe which resource is contained in each ordinary chest but cannot observe the content of the enchanted chest. At each episode the objectives contained in the enchanted chest are resampled. The optimal policy in this scenario is for (a_2) to collect the enchanted chest while the other agent (a_1) collect the chest containing the resource that was not in the enchanted chest, however this is only possible if a_1 knows what a_2 has collected. The observations of the agents are similar to the ones in the resource distribution environment at the exeception that agents now also observe the location of enchanted chest but cannot see their content before looting them.

B IMPLEMENTATION DETAILS

Experimental pipeline: We train the agents for 4 million timesteps in the resource distribution environment and 2 million timesteps in the multi-walker environment. The joint policy of the agents is periodically evaluated after 25,000 timesteps. Since the joint policies learned are usually stochastic each evaluation is conducted on 100 random environments. We log the median, first, and third quartile obtained from the 100 evaluations. To reduce the impact of random initialization on the neural network modelling the policy, the training is repeated on 5 random seeds.

Scalarization function for the multi-walker stability environment:A custom scalarization function is used instead of the standard Nash product because of the specific characteristics of the reward components: the second component r_2 is always negative, while the first component r_1 can be either positive or negative. In a traditional Nash product, one would compute the product $r_1 \cdot r_2$, which assumes that both components are strictly positive or at least consistently signed in a way that makes the product meaningful for optimization (e.g., maximizing the product implies improving both components). However, since r_2 is always negative, a straightforward product would always be negative and could lead to misleading gradients, especially when r_1 is positive (resulting in a more negative product). To address this, the scalarization explicitly checks whether either component is negative—penalizing the scalarization value when that is the case—by using the indicator $(r_1 < 0) \vee (r_2 < 0)$, and then computing

$$\text{scalarized\_reward} = \frac{-1 \cdot r_1 \cdot r_2}{1000}.$$

The division by 1000 serves as a normalization factor to keep the scalarized reward in a manageable numerical range. This adjusted scalarization function thus ensures more meaningful learning signals in environments where reward components have asymmetric signs.

Training hyperparmeters: Table 2 presents the key hyperparameters used in the implementation of the Decentralized MOCAC and Decentralized EUPG algorithms. The rest of the aglorithms were trained using the hyperparameters suggested in their respective papers.

Hyperparameter	Decentralized MOCAC	Decentralized EUPG
Number of atoms (n_{atoms})	11	–
Value range minimum (v_{min})	[0, 0, 0]	–
Value range maximum (v_{max})	[1, 1, 1]	–
Network width	64	64
Bootstrapping steps (n_{steps})	5	–
Learning rate (α)	5×10^{-4}	1×10^{-4}
Entropy coefficient (β)	1×10^{-3}	–

Table 2: Selected hyperparameters used in the decentralized MOCAC and EUPG algorithms. Dashes (–) indicate that the parameter does not apply to the corresponding algorithm.

C ALGORITHMS

This section presents how the algorithms proposed by [20, 21] were adapted to the cooperative multi-agent case.

Algorithm 2: Decentralized EUPG

Data:
$tr\_timesteps > 0$ // Num training timesteps
$\alpha > 0$ // Learning rate of the algorithm
$n\_agents > 0$ // Num. of agents
$u : \mathbb{R}^d \to \mathbb{R}$ // scalarisation function

Initialize the policy parameter θ^i for each agent i at random.;
$steps \leftarrow 0$;
while $steps < tr\_timesteps$ **do**
  Generate one episode by following the joint policy Π: $\mathbf{S}_0, \mathbf{A}_0, \mathbf{R}_0, \mathbf{S}_1, ..., \mathbf{S}_{T-1}, \mathbf{A}_{T-1}, \mathbf{R}_{T-1}, \mathbf{S}_T$;
  for *each agent* **do**
    for $t \in [1, T]$ **do**
      Estimate $\mathbf{G}_t^+$ and $\mathbf{G}_t^-$;
      Update the agent's policy parameters θ^i as
      $\theta^i \leftarrow \theta^i + \alpha\gamma^t u(\mathbf{G}_t^- + \mathbf{G}_t^+)\nabla_{\theta^i} \ln \pi_{\theta^i}(a_t^i \mid h_t^i, G_t^-)$;
    end
  end
  $steps \leftarrow steps + T$;
end

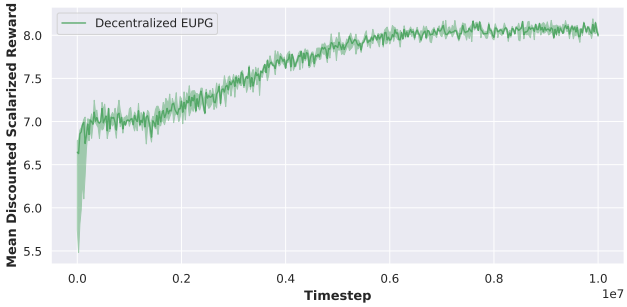
Algorithm 3: Decentralized Multi-Objective Categorical Actor-Critic

Data:
$tr\_timesteps > 0$// Num. of training timesteps
$\alpha_\pi > 0, \alpha_V > 0$ //
// learning rates of actor and critic
$n\_agents > 0$ // Number of agents
$u : \mathbb{R}^d \to \mathbb{R}$ // Scalarisation function
$[v_{\min}^{(j)}, v_{\max}^{(j)}]$ // Value range for each objective

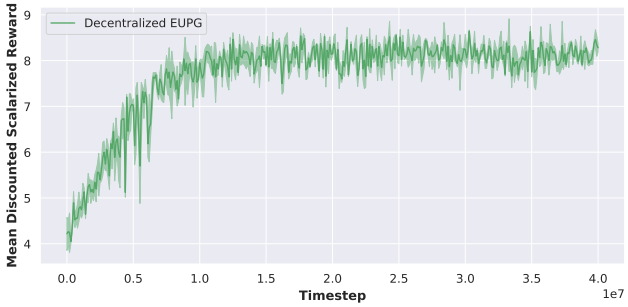
Initialize actor and critic parameters (θ^i, ϕ^i) for each agent i;
Define N atoms $\{z_k^{(j)}\}_{k=1}^N$ for each objective j using linear spacing in $[v_{\min}^{(j)}, v_{\max}^{(j)}]$;
$steps \leftarrow 0$;
while $steps < tr\_timesteps$ **do**
  Each agent i collects a trajectory $\tau_i = \{(t, s_t^i, \mathbf{r}_t^i, a_t^i, d_t^i, s_{t+1}^i)\}$ by acting in the environment using its policy π_{θ^i};
  foreach *transition* $(t, s_t^i, \mathbf{r}_t^i, a_t^i, d_t^i, s_{t+1}^i) \in \tau_i$ **do**
    Compute the projected target value distribution $\Phi\widehat{\mathcal{T}}Z_{\phi^i}(s_t^i)$
    Update critic by minimizing the cross-entropy between predicted distribution $Z_{\phi^i}(s_t^i)$ and target $\Phi\widehat{\mathcal{T}}Z_{\phi^i}(s_t^i)$ $\phi^i \leftarrow \phi^i - \nabla_{\phi^i}\alpha_V H(\Phi\widehat{\mathcal{T}}Z_{\phi^i}(s_t^i), Z_{\phi^i}(s_t^i))$
    Scalarize predicted value distribution: $Q_t^i = \sum_{k=1}^N u(z_k) \cdot p_{\phi^i}(z_k \mid s_t^i, a_t^i)$
    Update actor with policy gradient: $\theta^i \leftarrow \theta^i + \alpha_\pi \gamma^t Q_t^i \nabla_{\theta^i} \log \pi_{\theta^i}(a_t^i \mid s_t^i)$
  end
  $steps \leftarrow steps + |\tau_i|$ for all i;
end

D ADDITIONAL EXPERIMENTAL RESULTS

In this section we present the results obtained by our DEC-EUPG algorithm on the randomized households setting of the resource distribution task. Figure 5 presents the results of the DEC-EUPG algorithm on both the 2-agent 2-objective scenario and 4-agents 4-objectives scenario. We can see that on both tasks the agents are able to learn efficient policies. In the 2-agents bi-objective tasks the agents manage to learn a policy very close to the optimal policy getting a return of 8 when the optimal policy would get a an expected scalarized return of 9. In the 4 agents 4 objective task the policy learned is still interesting as it reaches a mean scalarized return of 8 but it shows that our method can still be improved to achieve better returns.



(a) RD:2O-2A



(b) RD:4O-4A

Figure 5: Results obtained by Dec-EUPG on the scaling experiments