Multiagent Credit Assignment for Multi-Objective Coordination

Raghav Thakar
thakarr@oregonstate.edu
Collaborative Robotics and Intelligent Systems Institute
Oregon State University
Corvallis, Oregon, USA

Gaurav Dixit
dixitg@oregonstate.edu
Collaborative Robotics and Intelligent Systems Institute
Oregon State University
Corvallis, Oregon, USA

Siddarth Iyer
viswansi@oregonstate.edu
Collaborative Robotics and Intelligent Systems Institute
Oregon State University
Corvallis, Oregon, USA

Kagan Tumer
kagan.tumer@oregonstate.edu
Collaborative Robotics and Intelligent Systems Institute
Oregon State University
Corvallis, Oregon, USA

Abstract

Many real-world coordination tasks—such as environmental monitoring, traffic management, and underwater exploration—are best modelled as multiagent problems with multiple, often conflicting objectives. Achieving effective coordination in these settings requires addressing two main challenges: 1) balancing multiple objectives and 2) resolving the credit assignment problem to isolate each agent's contribution from team-level feedback. Existing multiagent credit assignment methods collapse multi-objective reward vectors into a single scalar—potentially overlooking nuanced trade-offs. In this paper, we introduce the *Multi-Objective Difference Evaluation* (D_{MO}) operator to assign agent-level credit without a priori scalarisation. D_{MO} measures the change in hypervolume when an agent's policy is replaced by a counterfactual default, capturing how much that policy contributes to each objective and to the Pareto front. We embed D_{MO} into the popular NSGA-II algorithm to evolve a population of joint policies with distinct trade-offs. Empirical results on the Multi-Objective Beach Problem and the Multi-Objective Rover Exploration domain show that our approach matches or surpasses existing baselines, delivering up to a 33% performance improvement.

CCS Concepts

• **Computing methodologies** → **Cooperation and coordination**; **Multi-agent systems**; *Intelligent agents.*

Keywords

Multiagent Learning, Multi-Objective Evolution, Multiagent Credit Assignment

1 Introduction

Many real-world tasks such as environment monitoring [10], urban traffic management [19], and underwater exploration [41] are complex multiagent coordination problems. Learning to coordinate becomes particularly difficult when these problems contain multiple, possibly even conflicting objectives [39]. For instance, in an environmental monitoring task, these may include 1) evenly covering the environment, 2) performing focused monitoring of specific points of interest, while 3) minimising overall energy expenditure. Success in these settings requires learning coordinated multiagent joint policies that optimise multiple objectives at once.

There are two key challenges in complex multi-objective multiagent learning problems: 1) balancing multiple objectives to learn rich trade-offs, and 2) discerning feedback for individual agents from an all-encompassing team reward, i.e. the *multiagent credit assignment* problem. The multiagent credit assignment problem is particularly challenging in multi-objective domains, wherein an agent's impact must be measured across multiple objectives simultaneously.

Many solutions for the multiagent credit assignment problem have been developed for single-objective Multiagent Reinforcement Learning (MARL) [4, 15, 29] and Cooperative Coevolution [1, 6, 8], and have shown to significantly aid the learning process. In multi-objective settings, these methods are extended via a priori *weighted scalarisation* that collapses the multi-objective reward vector into a single scalar value [26, 42].

A drawback to a priori scalarisation is that it imposes a fixed preference, and may overlook complex, non-linear trade-offs among the objectives. This can severely limit scalarisation-based approaches from fully capturing the solution space and learning the true Pareto front [11, 21, 40]. Several Multi-Objective Evolutionary Algorithms (MOEAs) [9, 14, 45] provide a scalarisation-free alternative to learning the Pareto front. However, their application in multiagent learning remains limited without addressing the multi-objective multiagent credit assignment problem.

In this work, we introduce the Multi-Objective Difference Evaluation (D_{MO}) operator for estimating agent-level credit in multi-objective evolutionary algorithms. D_{MO} measures the change in hypervolume when an agent's policy is replaced by a *counterfactual* default, effectively capturing how much that policy contributes to the Pareto front [30]. This measure is then used as the agent's credit value.

Our key insight is that 'contribution to the hypervolume' is an effective agent-level feedback to learn from, and allows MOEAs to evolve a population of joint policies that expresses distinct trade-offs among the objectives. D_{MO} requires no a priori reward scalarisation[1], which 1) enables learning without needing the expertise to

[1]While the hypervolume metric does produce a scalar value, it is *not* used here to produce a singular scalar reward before learning. Instead, we apply it *within* an existing

design complex scalarising functions, and 2) promotes an expansive search of the objective space.

Our primary contribution in this paper is the D_{MO} operator. We leverage the D_{MO} operator by minimally modifying the NSGA-II algorithm [14]. We then compare this modified NSGA-II with a coevolutionary approach from the literature that combines NSGA-II with Difference Evaluation for credit assignment, classical NSGA-II, and modified NSGA-II without credit assignment. We match the performance of existing baselines in the Multi-Objective Beach Problem, and show a 33% increase in performance in the Multi-Objective Rover Exploration Problem.

2 Background

2.1 Multi-Objective Optimisation

Many real-world problems are multi-objective, where improving in one objective is detrimental to another (e.g., speed vs. safety vs. comfort in autonomous vehicles [23]). Instead of a single *best* solution, it is often preferred to develop a *range* of Pareto-optimal solutions that provide different trade-offs across objectives.

A majority of the work in recent years falls into one of two categories—methods that focus on 1) learning a Pareto front estimate directly [9, 14, 20, 22, 45], and 2) optimising a single *super* objective created by applying a scalarising or utility function over all the objectives [3, 24, 26, 28]. For multiagent learning, agent-specific utility functions are generally favoured [33]. However, collapsing a multi-objective fitness vector into a scalar value risks imposing sub-optimal preferences, particularly when objectives are complexly interdependent [11, 21, 40]. Additionally, preferences among objectives to design these utility functions may not exist beforehand. Lastly, in critical applications, decision-makers may prefer choosing from a range of Pareto-optimal solutions rather than optimising predefined preferences. Thus, it is often desirable to operate in the *decision support* [33] paradigm, which involves learning an estimate of the whole Pareto front to provide decision-making support, instead of optimising predefined utilities over the objectives.

Multi-Objective Evolution. Many MOEAs, such as PAES [22], PESA-II [9], NPGA [20], SPEA2 [45] and NSGA-II [14], provide a scalarisation-free approach to multi-objective learning when utilities are unknown. Another advantage of using MOEAs (and evolutionary methods in general) is their indifference to the frequency of the availability of feedback within an episode. Any feedback is only utilised at the *end* of an episode, in the form of the candidate solution's *fitness*, computed by simply aggregating whatever little feedback the environment provides. This makes evolution especially effective in *sparse reward* settings, where the lack of dense feedback makes extracting a gradient for gradient-based learning challenging [36, 37]. Hence, we focus on population-based MOEAs, and specifically, NSGA-II [14]. The NSGA-II algorithm remains the most popular for problems with few objectives and serves as the most suitable for comparison. In this paper, NSGA-II and its operators will be a common recurrence.

scalarisation-free MOEA strictly for agent-level credit assignment, thus preserving the benefits of scalarisation-free methods.

2.2 Multiagent Systems and the Credit Assignment Problem

One of the key problems in multiagent systems research is the credit assignment problem, where the effect of an agent's actions on the team fitness must be determined. This quantified contribution is then used as feedback for the agent, promoting positive, team-oriented actions. Credit can be provided in several ways, including as a reward in (deep) reinforcement learning [17, 29], or as a local fitness to determine the selection probabilities of policies in evolutionary algorithms [6, 7]. Credit assignment becomes important when the team fitness is too general for individual agents to efficiently learn impactful behaviours from.

Difference Evaluation. The Difference Evaluation operator (D) is a state-of-the-art technique that addresses the credit assignment problem by estimating the contribution of a single agent to a multiagent team's performance [6, 15]. For an *agent i*, the Difference Evaluation is defined as:

$$D_i = G(\mathbf{z}) - G(\mathbf{z_{-i}} \cup \mathbf{c_i}), [34] \tag{1}$$

where $\mathbf{z}$ is the joint-action of the system, $G(\mathbf{z})$ is the global system performance, $\mathbf{z_{-i}}$ is the joint-action of the system with the action of agent i removed, and $\mathbf{c_i}$ is a *counterfactual* action that agent i's action is replaced with. $G(\mathbf{z_{-i}} \cup \mathbf{c_i})$ gives an estimate of the performance of a hypothetical system without the contribution of agent i. The *counterfactual* term $\mathbf{c_i}$ represents a *default* action with no contribution to the system performance. This default action comes intuitively in many problems, such as sticking to a starting position indefinitely in an environment-exploration problem [30].

D has been successfully employed in evolving a multiagent joint policy using Cooperative Coevolutionary Algorithms (CCEAs), providing local fitness evaluations to guide the policy evolution of each individual agent [1, 6, 8]. In Multiagent Reinforcement Learning (MARL) contexts, D is leveraged to shape highly targeted rewards for each agent [4, 15].

Interestingly, D has been employed for multi-objective multiagent credit assignment in prior works [42, 43]. In the MARL approach of these works, each agent's contribution to each objective is estimated via D, then combined with a scalarising function, making credit assignment contingent on predetermined objective preferences and placing the method within the scalarisation-based paradigm. The authors of these works also propose a coevolutionary approach that evolves each agent's policies in separate subpopulations, using NSGA-II on local fitnesses computed using D. As this approach is scalarisation- and utility-free, it will serve as a baseline to compare our work with.

3 Method

In this section we define the D_{MO} operator and describe its incorporation into NSGA-II.

3.1 The Multi-Objective Difference Evaluation Operator

Evolutionary Algorithms (EAs) maintain a population of candidate solutions, each representing a parametrised entity under optimisation. For multiagent applications, each candidate solution can encode a joint policy—a collection of single agent policies—evaluated using a multi-objective fitness vector. However, using only the fitness vector—which evaluates the entire team's performance—overlooks the quality of individual agent policies. Highly impactful single-agent policies in underperforming joint policies may be discarded, while low-impact ones in strong joint policies can survive, reducing evolutionary efficiency. This scenario exemplifies the multi-objective multiagent credit assignment problem, which calls to determine agent-level contributions to joint policy performance.

'Joint policy performance' should not only reflect the joint policy's objective-wise performance, but also its contribution to the Pareto front's spread, and its uniqueness within the population. Our key insight is that the *hypervolume* of a nondominated set naturally captures these factors [18, 35, 44], and a single agent policy's contribution to the hypervolume is impactful agent-level credit.

We thus propose the Multi-Objective Difference Evaluation (D_{MO}) operator. D_{MO} replaces one agent's trajectory (i.e. state-action pairs collected from interactions with the environment) with a counterfactual default-action trajectory and measures the resulting change in hypervolume. Thus, D_{MO} isolates the single agent policy's contribution to its joint policy's performance.

Notation and Definitions

- Let $\mathcal{J}$ be the nondominated set of joint policies.
- Each joint policy $\boldsymbol{\pi} \in \mathcal{J}$ consists of k individual policies:
$$\boldsymbol{\pi} = (\pi_1, \pi_2, \ldots, \pi_k).$$
- Each policy π_i generates a trajectory:
$$\tau_i = \tau(\pi_i).$$
- The joint trajectory generated by $\boldsymbol{\pi}$ is:
$$\boldsymbol{\tau} = (\tau_1, \tau_2, \ldots, \tau_k).$$
- The set of joint trajectories corresponding to $\mathcal{J}$ is:
$$\mathcal{T} = \{\boldsymbol{\tau} \mid \boldsymbol{\tau} \text{ is generated by } \boldsymbol{\pi} \in \mathcal{J}\}.$$
- Let $H(\mathcal{T})$ denote the hypervolume of the set $\mathcal{T}$:
$$H(\mathcal{T}) = \text{hypervolume}\left(\{\text{fitness}(\boldsymbol{\tau}) \mid \boldsymbol{\tau} \in \mathcal{T}\}\right).$$

$\mathrm{D_{MO}}$ Computation

We assess the impact of an individual policy by replacing its trajectory with a counterfactual trajectory consisting of default actions.

(1) **Counterfactual Trajectory Replacement**
- For the policy π_i under consideration, define a default policy d_i that generates a default trajectory:
$$c_i = \tau(d_i).$$
- Construct the counterfactual joint trajectory by replacing τ_i with c_i:
$$\boldsymbol{\tau}' = (\tau_1, \ldots, \tau_{i-1}, c_i, \tau_{i+1}, \ldots, \tau_k).$$

(2) **Modified Trajectory Set**

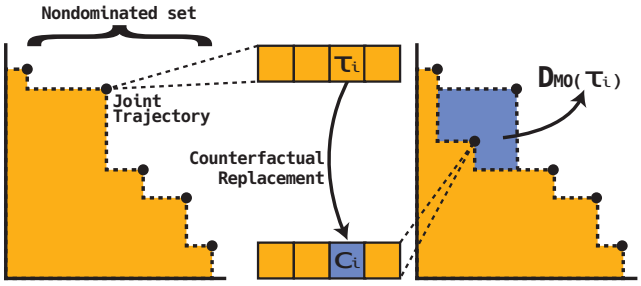


Figure 1: Graphical representation of the D_{MO} computation procedure. From a joint trajectory, an agent's trajectory is swapped with a counterfactual default, and the subsequent change in hypervolume is assigned as the agent's credit.

- Obtain the modified set by replacing $\boldsymbol{\tau}$ with $\boldsymbol{\tau}'$ in $\mathcal{T}$:
$$\mathcal{T}' = (\mathcal{T} \setminus \{\boldsymbol{\tau}\}) \cup \{\boldsymbol{\tau}'\}.$$

(3) **$\mathrm{D_{MO}}$ Value Calculation**
- Compute the D_{MO} value to quantify the impact of the individual policy π_i:
$$D_{MO}(\pi_i, \boldsymbol{\pi}, \mathcal{J}) = H(\mathcal{T}) - H(\mathcal{T}'). \quad (2)$$

Figure 1 provides an intuitive visual of this D_{MO} computation. A key requirement is that each objective must be of the same nature. Either all must be *maximisation* objectives, or all must be *minimisation*. Assuming maximisation across all objectives, a positive D_{MO} value implies that the hypervolume reduces when τ_i is replaced by a counterfactual c_i. The higher the D_{MO} value for a policy, the more *impactful* it can be considered, and the more likely should be its preservation and proliferation in the evolution process. On the flip side, D_{MO} values are also useful for identifying poor-performing single agent policies and actively discouraging their insidious propagation over generations. Weeding out bad policies is as important as preserving good ones, as it 'frees' up the corresponding agent to explore other behaviours. In settings with easily accessible local optima in team behaviour, this additional exploration is key for the team to achieve globally optimal performance.

Algorithm 1 `assign-DMO`($\mathcal{J}$)

Require: Nondominated set of joint policies $\mathcal{J}$
1: Let $\mathcal{T} = \{\boldsymbol{\tau} \mid \boldsymbol{\tau}$ is generated by $\boldsymbol{\pi} \in \mathcal{J}\}$ ▷ Joint trajectory set
2: **for all** joint policies $\boldsymbol{\pi} \in \mathcal{J}$ **do**
3: Let $\boldsymbol{\pi}$ contain the policy vector $[\pi_1, \pi_2, \ldots, \pi_k]$
4: Let $\boldsymbol{\tau} = \{\tau_1, \tau_2, \ldots, \tau_k\}$ be the joint trajectory $\boldsymbol{\pi}$ generates
5: **for all** $i \in |\boldsymbol{\pi}|$ **do** ▷ Note that $|\boldsymbol{\pi}| = |\boldsymbol{\tau}|$
6: Let d_i be the default policy for agent i
7: $c_i = \tau(d_i)$ ▷ Generate counterfactual trajectory
8: $\boldsymbol{\tau}' \leftarrow (\boldsymbol{\tau} \setminus \{\tau_i\}) \cup \{c_i\}$ ▷ Replace trajectory
9: $\mathcal{T}' = (\mathcal{T} \setminus \{\boldsymbol{\tau}\}) \cup \{\boldsymbol{\tau}'\}$ ▷ Replace joint trajectory
10: π_i.`DMOValue` $\leftarrow H(\mathcal{T}) - H(\mathcal{T}')$ ▷ Equation 2
11: **end for**
12: **end for**

3.2 Modifying Classical NSGA-II to Leverage Policy-Level Credit

We present a minimally modified version of classical (real-coded) NSGA-II that leverages D_{MO} values in the selection and crossover operations to create the offspring set. We choose the NSGA-II algorithm for its robustness and widespread adoption across a range of applications [16, 27]. We modify classical NSGA-II to be able to utilise fitness values local to each policy in the joint policies being evolved.

We introduce a new step, `assign-DMO`, detailed in **Algorithm 1**, to compute and assign policy-level credit. Once these policy-level credit values have been obtained, we modify the `make-new-pop` procedure of NSGA-II to incorporate D_{MO} credit value in parent selection and crossover. Unlike classical NSGA-II, where two complete parent joint policies would be selected via binary tournaments and crossed over using the Simulated Binary Crossover (SBX) [13], we conduct the binary tournament selection and subsequent SBX crossover *per single agent policy* rather than *per joint policy*. This produces two offspring solutions, and this process is repeated until the offspring set is full. Specifically, for each policy slot, the tournament compares the D_{MO} credit value of candidate parent single policies and selects two parents. We then apply SBX on this pair of single policies, repeating for all policy indices. One pass of this procedure yields two complete offspring solutions. We then repeat this until the offspring set is full.

By substituting whole-individual selection with policy-level selection, we minimally extend NSGA-II so that local fitnesses, like D_{MO} values, guide the offspring creation process. As each constituent policy of an offspring solution is produced from two parent single policies, for an offspring solution of size K, there may be a maximum of $2K$ parent solutions that are used to create it. We formally define this modified `make-new-pop` procedure in **Algorithm 2**.

Algorithm 2 `make-new-pop`(P)

Require: Parent set P of joint policies
Ensure: Offspring set Q

$Q \leftarrow \emptyset$ ▷ Initialise empty offspring population
Let N be the desired size of the offspring set
Let K be the number of single policies in each joint policy
while $|Q| < N$ **do**
  $\boldsymbol{\pi}_{\text{offspring1}} = \{\emptyset_1, \emptyset_2, \ldots, \emptyset_K\}$ ▷ Initialise blank joint policy
  $\boldsymbol{\pi}_{\text{offspring2}} = \{\emptyset_1, \emptyset_2, \ldots, \emptyset_K\}$ ▷ Initialise blank joint policy
  for $k = 1$ **to** K **do**
    $\boldsymbol{\pi}_{\text{parents}} \leftarrow \{\boldsymbol{\pi}[k] \mid \boldsymbol{\pi} \in P\}$ ▷ Candidate parents
    $\pi_{\text{p1}}, \pi_{\text{p2}} \leftarrow \text{select}(\boldsymbol{\pi}_{\text{parents}})$ ▷ Using D_{MO} values
    $\pi_{o1}, \pi_{o2} \leftarrow \text{SBX}(\pi_{p1}, \pi_{p2})$ ▷ Crossover
    $\boldsymbol{\pi}_{\text{offspring1}}[k] \leftarrow \pi_{o1}$ ▷ k^{th} policy of offspring 1
    $\boldsymbol{\pi}_{\text{offspring2}}[k] \leftarrow \pi_{o2}$ ▷ k^{th} policy of offspring 2
  end for ▷ Two offsprings have been created
  $\boldsymbol{\pi}_{\text{offspring1}} \leftarrow \texttt{mutate}(\boldsymbol{\pi}_{\text{offspring1}})$
  $\boldsymbol{\pi}_{\text{offspring2}} \leftarrow \texttt{mutate}(\boldsymbol{\pi}_{\text{offspring2}})$
  $Q \leftarrow Q \cup \{\boldsymbol{\pi}_{\text{offspring1}}, \boldsymbol{\pi}_{\text{offspring2}}\}$
end while
return Q

4 Testing Domains

In this section, we describe two multiagent coordination problems that require a team of agents to balance multiple objectives and provide rich trade-offs among them. Learning is complicated due to only the *team reward* being available, which necessitates agents to receive more accurate, personalised feedback to be able to learn coordination efficiently. Thus, not only do these domains test the ability of learning methods to provide Pareto-optimal trade-offs, but also investigate the value of credit assignment in efficiently learning coordination in complex multi-objective problems.

4.1 The Multi-Objective Beach Problem (Beach) Domain

The Beach domain is a multi-objective partially observable Stochastic Game that has been developed as a benchmark problem domain for multiagent learning algorithms [25]. It is an extension of the Multi-Objective Bar Problem [42], and the El Farol Bar Problem [2].

Agents (tourists) are distributed across several sections of a beach and must decide whether to stay in their current section or move to an adjacent one; the actions being `move_left`, `stay`, and `move_right`. Each agent is only provided knowledge of the beach section it starts from. The goal is to optimise two conflicting objectives: (1) *capacity*, which is maximised when the number of agents in a section matches its ideal capacity, and (2) *mixture*, which is maximised when there is an equal number of the two agent types (e.g., introverts and extroverts) in each section.

We now define the two objectives of this problem. The global capacity objective is to maximise G_{cap}, the sum of the local capacity rewards over all sections:

$$G_{cap} = \sum_{s \in S} L_{cap}(s)$$

where the local capacity reward, $L_{cap}(s)$, for a given section s is calculated as:

$$L_{cap}(s) = x_s e^{-\frac{x_s}{\psi}}$$

Here, x_s is the number of agents in section s and ψ is the ideal capacity of that section.

The global mixture objective is to maximise G_{mix}, the sum of the local mixture rewards over all sections:

$$G_{mix} = \sum_{s \in S} L_{mix}(s)$$

where the local mixture reward, $L_{mix}(s)$, for a given section is:

$$L_{mix}(s) = \frac{\min(|I_s|, |E_s|)}{(|I_s| + |E_s|) \times |S|}$$

Here, $|I_s|$ and $|E_s|$ are the number of agents of each type in section s, and $|S|$ is the total number of sections.

Thus, the global reward vector G for an episode is given by:

$$G = \left[G_{cap}, G_{mix}\right]$$

There are two important features of the rewards in this domain. First, the environment only returns the net global reward vector to each agent. Thus, each agent must learn *individual* actions from the multi-objective *team* reward vector. Second, this domain yields a non-zero reward on each objective for every possible distribution of agents across beach sections. Thus, the rewards in this domain

are *dense*, and even slight changes in the distribution of agents causes corresponding changes in the rewards. The tight correlation between the actions agents take and the feedback they receive weakens the need for sophisticated multiagent credit assignment. However, we still consider experiments in this domain necessary for benchmarking.

4.2 The Multi-Objective Rover Exploration Problem (Rover) Domain

The Rover domain is a multi-objective extension of a classic multiagent coordination problem [5, 6, 34]. It serves as a proxy for real-world tasks like environmental monitoring [31], underwater exploration [41], and distributed lunar sensing [12]. Due to partial observability, this problem may be modelled as a Multi-Objective Partially Observable Markov Decision Process.

A team of homogeneous rovers operates on a 2D plane to *observe* Points of Interest (POIs), each of which provides rewards on one or more objectives. Agents must coordinate navigation and learn the *trade-offs* among objectives by prioritising different POIs. Each rover's observation state averages $\exp(-d)$ measurements for nearby agents and POIs in distinct channels (one for agents and separate ones for each objective), subdivided into multiple sectors to enable more refined decision-making. Given these observations, each agent outputs navigational actions (d_x, d_y), subject to a maximum step length $|L|$.

Some POIs have *coupling* requirements, requiring simultaneous observation by multiple rovers to yield any reward. Thus, the learning challenges are threefold:

- Navigating to POIs,
- Coordinating to satisfy coupling constraints,
- Developing a suite of behaviours that provide distinct trade-offs across objectives.

Rewards are *sparse* and gained only when a POI's coupling requirement is fulfilled. Variations in POI objectives, coupling constraints, reward magnitudes, and required proximity render the Rover domain a challenging multi-objective multiagent coordination problem.

5 Experiments

5.1 The Algorithms

To study the impact of credit assignment in D_{MO}, we compare the D_{MO}-incorporated NSGA-II algorithm (Section 3.2) (D_{MO} hereon) with three baselines.

(1) **Classical NSGA-II (NSGAII):** The unmodified real-coded version of NSGA-II as originally published [14].
(2) **Decentralised NSGA-II with Credit Assignment (NSGAII+CA):** A coevolutionary approach where each agent's policies are evolved in separate subpopulations [43]. It uses **D** to derive a multi-objective credit vector for each policy from the multi-objective team reward vector.
(3) **NSGA-II with Policy-Level Selection and Crossover (NSGAII+PLSC):** An ablated baseline that retains the single-agent policy structure of D_{MO}-incorporated NSGA-II but does not use policy-level credit. Instead, each parent single-agent policy is selected based on the Pareto dominance and crowding distance of the *joint policy* it belongs to.

In each experiment, policies are neural networks with parameters randomly initialized in $[-1, 1]$. For a population of size N, we retain the top $N/2$ solutions each generation and create $N/2$ offsprings. We mutate each offspring by applying Gaussian noise (mean= 0, standard deviation= 0.5) to each network parameter with probability $\phi = 0.75$. Crossover is performed using the SBX operator with a distribution index $\eta = 15$. For counterfactual replacements in D_{MO} and NSGAII+CA, we use a *null* trajectory that removes an agent entirely from the joint trajectory.

5.2 Beach Domain Experiments

We set up two instances of the problem: $\texttt{Beach}_{50}$ and $\texttt{Beach}_{100}$, with 50 and 100 agents respectively. Each instance has five beach sections. Each agent makes one move—`move_left`, `move_right`, or `stay`. An agent receives the ID of the section it occupies as state input in a one-hot encoded format. In each instance, 70% of the agents are of Type `I`, while 30% are of Type `E`. For each instance, we assign `section 2` as the starting section for half of the Type `I`, and half of the Type `E` agents. For the remaining agents of each type, we assign the starting section as `section 4`[2]. For $\texttt{Beach}_{50}$, we set each beach section's capacity $\psi = 3$ and for $\texttt{Beach}_{100}$, we set the capacity $\psi = 5$.

Pareto-optimal solutions in this domain are attained when most agents crowd one beach section. In each remaining beach section, agents must either occupy the section to match its capacity exactly, or ensure that equal number of both agent types occupy the section. Due to the section capacities being odd values, at best, only one of the two objectives may be maximised from each beach section. This predicament is exacerbated by the imbalance in the proportion of the two agent types in the system. Thus, the goal is to learn a suite of joint policies that express all possible Pareto-optimal trade-offs by attaining the various optimal distributions of agents across beach sections.

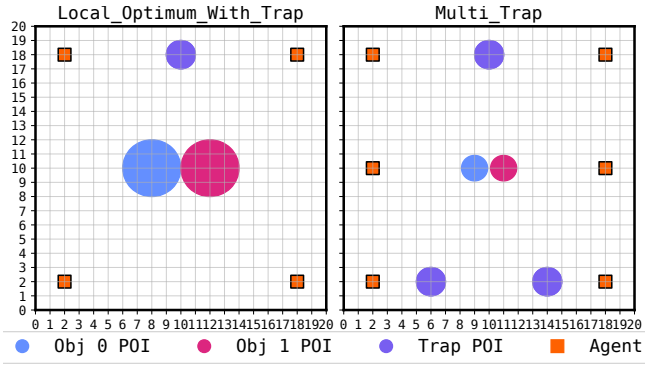
We test each method in each instance with a population size $N = 100$ and ten statistical runs with random seed $\lambda \in \{2024 \cdots 2033\}$.

5.3 Rover Domain Experiments

In the Rover domain, we set up three experiments that each test multiagent coordination, and the impact of multiagent credit assignment on learning. In each set-up, the map is of size 20×20, agents have a maximum step length of 1 unit, agents can observe features in the map (like POIs and other agents) up to 5 units away, and the length of the episode is set at 25 timesteps. For Asymmetric Exploration, we set the population size $N = 400$, for Local Optimum, we set $N = 200$, and for Multi-Trap, $N = 300$. We perform five trials, with random seed $\lambda \in \{2024, 2025, \cdots 2028\}$.

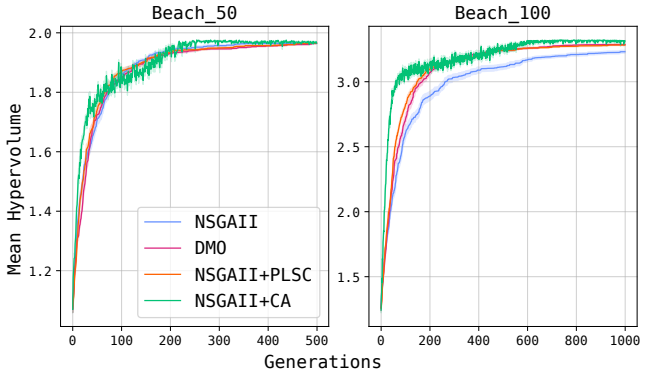
5.3.1 ***Asymmetric Exploration.*** In this setup, a team of eight agents starts at the map's geometric centre. Eight identical POIs, each with coupling=2, are uniformly placed around the centre and grant a single +2 reward when simultaneously observed by two agents (coupling=2). Once observed, these POIs do not yield further

[2]Starting beach sections are assigned from beach sections 1-5.



(a) Local Optimum (With Trap) (b) Multi-Trap

Figure 2: Environment maps for (a) Local Optimum (With Trap), and (b) Multi-Trap Rover domain problems. Each central POI (coupling=3) rewards on a *single*, distinct objective, while each trap POI (coupling=1) rewards on *both* objectives.



(a) $\texttt{Beach}_{50}$ Results (b) $\texttt{Beach}_{100}$ Results

Figure 3: Mean hypervolume comparison in the (a) $\texttt{Beach}_{50}$ and (b) $\texttt{Beach}_{100}$ domains. The rich feedback from the environment weakens the need for sophisticated multiagent credit assignment techniques and we see comparable performance across methods.

rewards, and maximising this reward is the first objective. Four additional coupling=2 POIs in each corner offer repeating +0.25 rewards, constituting the second objective. We run two configurations: one with the repeating POIs always active, and another where they are ephemeral, with each ephemeral POI having a distinct, five-timestep observation window. In these two configurations, we test the general-purpose robustness of learning methods. Team synergy is key, yet there are various strategies to maximise rewards and trade-offs. Lastly, we test how various methods fare with an increase in the problem difficulty, as introduced by the ephemeral POIs.

5.3.2 ***Local Optimum.*** We place four agents in the four corners of the map, with two adjacent radius=2 POIs[3] at the centre, each with coupling=3. Maximising the repeating +1 rewards from these two POIs forms the two objectives, but since they do not overlap, the team can observe only one POI at a time. Thus, agents must coordinate both navigation and the decision of which POI to observe, and for how long. Achieving rich trade-offs across the objectives requires tight coordination to balance and fully exploit POIs.

To further challenge the agents, we introduce a "trap" POI (coupling=1) that provides repeating rewards of +0.25 on both objectives. Low coupling makes this POI easier to discover, creating a local optimum. Pareto-optimal solutions are found when only one agent exploits the trap while the other three focus on the higher-value central POIs, balancing the two objectives. We run two configurations of this local optimum problem—one with the trap POI and one without it—to test whether learning methods can avoid this local optimum in favour of the global optimum. Figure 2a provides a visualisation of the map for this problem.

5.3.3 ***Multi-Trap.*** As a special test, we run a configuration similar to Local Optimum, but with three traps, each of coupling=1. We test this map with six agents, with three agents starting from either ends of the map. To preserve the challenge of discovering the central POIs with an increase in agents, we shrink the central POIs to radius=1. Figure 2b visualises the map configuration for this problem.

[3]The 'radius' is the maximum distance from which a POI may be observed by an agent

6 Results and Analysis

We now present and discuss our findings from the experiments described in Section 5. For each mean hypervolume chart, we plot the mean of the hypervolume attained over the respective trials, with the Standard Error of Mean (SEM) shaded in a lighter colour.

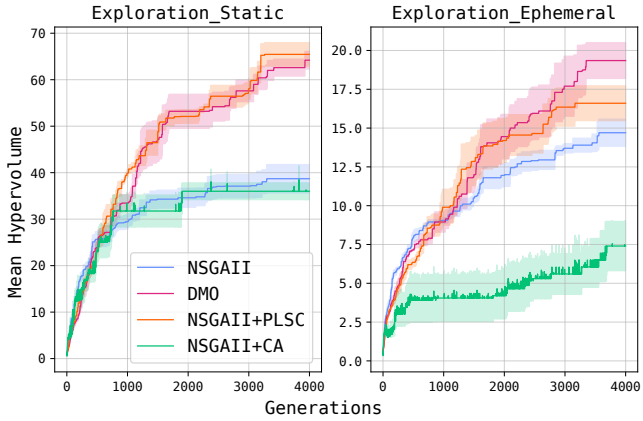
6.1 Beach Domain

Figure 3 compares the hypervolume attained by each method in the $\texttt{Beach}_{50}$ (Figure 3a) and $\texttt{Beach}_{100}$ (Figure 3b) problems. The hypervolume values attained are comparable across methods, with no discernible difference among D_{MO}, NSGAII+CA, and NSGAII+PLSC on either problems. NSGAII is also competetive in the $\texttt{Beach}_{50}$ problem. However, we notice a small dip in NSGAII in $\texttt{Beach}_{100}$, with NSGAII's final mean hypervolume being $\approx$ 2% lower than that of other methods.

With the feedback in the Beach domain being dense, even small changes in a single agent's decision causes a corresponding shift in the team reward. This correlation means that high-impact individual policies naturally proliferate even through naive selection and crossover based solely on the team fitness—explaining comparable performance across methods. Meanwhile, explicitly addressing credit assignment—as in D_{MO} and NSGAII+CA—introduces no adverse effects.

6.2 Rover Domain

6.2.1 ***Asymmetric Exploration.*** Figure 4 shows the mean hypervolume achieved by each method in the Asymmetric Exploration problem, both without and with ephemeral POIs. We see a clear advantage in selecting and crossing over single agent policies rather than complete joint policies, as evidenced by the higher performance of D_{MO} and NSGAII+PLSC in Figures 4a and 4b. D_{MO}



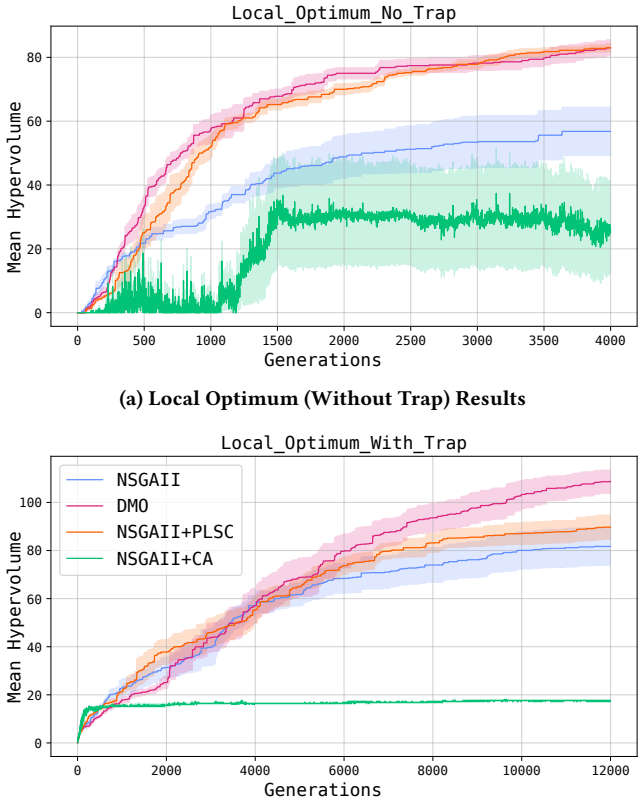
(a) Asymmetric Exploration (Static POIs) Results **(b) Asymmetric Exploration (Ephemeral POIs) Results**

Figure 4: Mean hypervolume comparison in the Asymmetric Exploration problem in the Rover domain with (a) static and (b) ephemeral POIs. An increase in difficulty caused by ephemeral POIs reveals the value of accurate agent-level feedback for learning efficiently.

outperforms NSGAII+PLSC by ≈ 15% in the ephemeral-POI version (Figure 4b), due entirely to accurate agent-level credit. As the problem grows more difficult, this informative feedback becomes increasingly critical.

We would also like to highlight a critical shortcoming of NSGA-II+CA, which uses a CCEA that randomly selects policies from each subpopulation to form a multiagent team policy to evaluate. This prevents single agent policies from consistently pairing with the same teammates from other subpopulations and thus inhibits complementary behaviours. This is a major hindrance to performing well in settings such as the Rover domain, which require agents to tightly coordinate to handle coupling constraints. Additionally, NSGA-II's focus on generating diverse solutions contradicts the coevolutionary need for subpopulations to converge, further undermining performance. These issues explain NSGA-II+CA's poor results in the Rover domain.

6.2.2 ***Local Optimum.*** Figure 5 compares the mean hypervolume each method attains in the Local Optimum problem without and with the trap POI. Overcoming a local optimum is a classic case of requiring the contribution of each agent to be measured clearly, so that underperforming agents can be identified and pushed to explore other useful behaviours [32, 38]. This is clearly shown in Figure 5b, where D_{MO} shines in allowing the agents to handle the trap POI and learn higher-hypervolume Pareto fronts. D_{MO}'s mean hypervolume is ≈ 20% higher than that of NSGAII+PLSC, and ≈ 33% higher than that of NSGAII. In Figure 6, we compare the Pareto fronts of performance on each objective attained by each method. Specifically, we compare Pareto fronts at the same percentile (by hypervolume) across the various trials of each method. As evident, in most trials, D_{MO} produces dominant, yet diverse solutions that offer rich and desirable trade-offs among objectives.



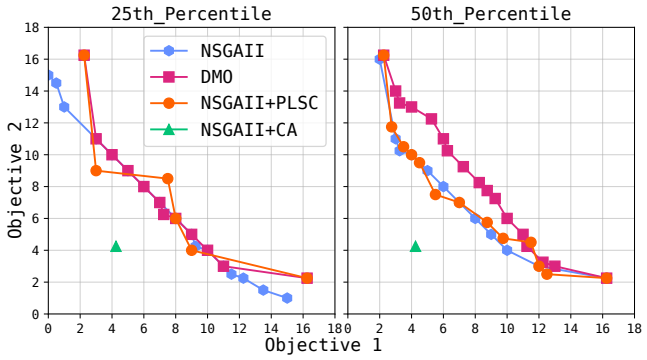
(a) Local Optimum (Without Trap) Results

(b) Local Optimum (With Trap) Results

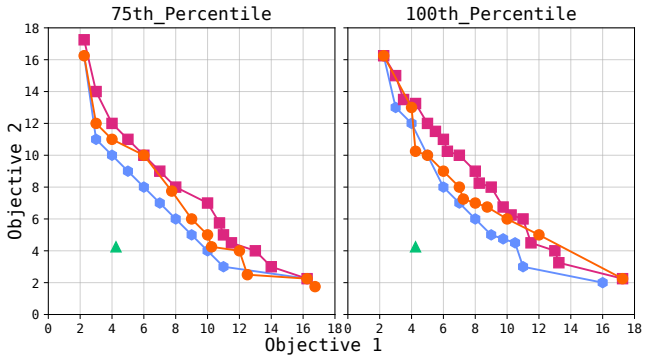
Figure 5: Mean hypervolume comparison in the Local Optimum problem in the Rover domain (a) without the trap POI, and (b) with the trap POI. Overcoming locally optimal behaviour is encouraged with accurate credit assignment, which 'calls out' single agent policies that do not contribute much to the team reward—compelling them to explore other behaviours instead.

6.2.3 ***Multi-Trap.*** Our next comparison, shown in Figure 7, focuses on the mean hypervolume of the fronts learned in the Multi-Trap problem. The three trap POIs here present an exceptional challenge in discovering the high-reward central POIs. D_{MO} is the first to surmount the local minima, as evidenced by a jump in mean hypervolume at around 20,000 generations. Once agents overcome the trap and locate higher-reward POIs, D_{MO} isolates and reinforces the high-performing policies, further improving globally beneficial behaviours. Accurate credit assignment thus proves critical for both—discovering rewarding strategies, and refining them afterwards. This also explains the steady climb in mean hypervolume D_{MO} sustains after overcoming the trap POIs in Figure 7.

As a final comparison, in Figure 8, we sample the median Pareto-fronts learnt by D_{MO} and NSGAII+PLSC, and plot the trajectories that demonstrated the most balanced trade-offs. We clearly see that D_{MO} is able to optimally exploit both—the trap POIs, and the central POIs while NSGAII+PLSC fails to surmount the trap POIs.



(a) 25th %ile Pareto fronts (b) 50th %ile Pareto fronts



(c) 75th %ile Pareto fronts (d) 100th %ile Pareto fronts

Figure 6: Comparison of the Pareto fronts at various percentiles (by hypervolume) learnt by each method in the Local Optimum problem (with trap) of the Rover domain. D_{MO} reliably learns dominant fronts that also consistently provide more trade-offs across objectives.

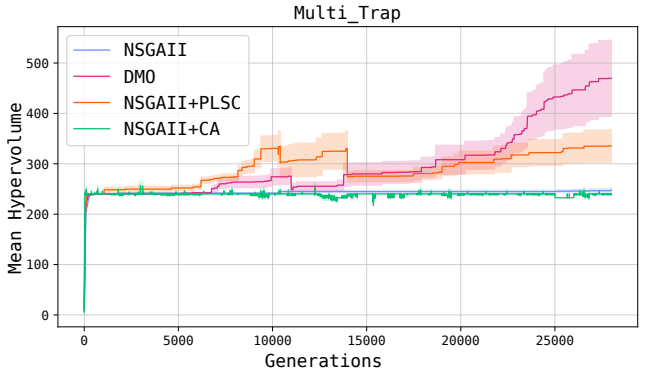
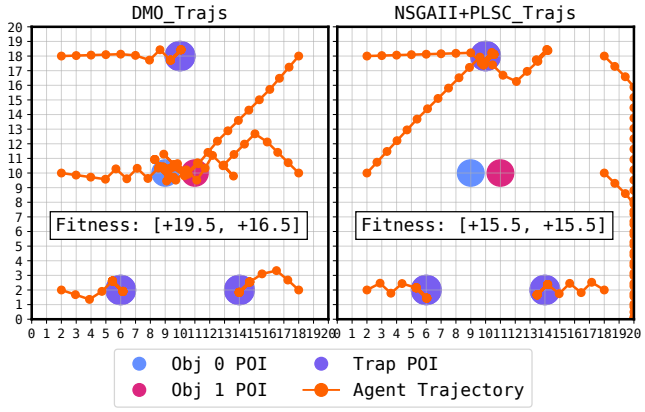


Figure 7: Extremely tight coordination is required to overcome three traps in the Multi-Trap Rover domain problem. D_{MO} is the first method to learn to reliably overcome the traps, as seen by the mean hypervolume jump starting at roughly 20,000 generations.



(a) Team trajectories displayed by D_{MO} (b) Team trajectories displayed by NSGAII+PLSC

Figure 8: Team trajectories that attained the most balanced Pareto-optimal trade-offs by (a) D_{MO} and (b) NSGAII+PLSC in the Multi-Trap problem.

7 Conclusion and Future Work

We presented the D_{MO} operator, a solution to the multiagent credit assignment problem for multi-objective coordination. We formally described D_{MO}, and then leveraged it in an NSGAII-like algorithm. Our results show competitive performance compared to baselines in the dense-reward Multi-Objective Beach Problem Domain, and D_{MO} outperforming the existing baselines by up to 33% in the sparse-reward Multi-Objective Rover Exploration Problem Domain. D_{MO} is able to consistently provide rich and dominant trade-offs without requiring any a priori scalarisation of the objectives. D_{MO} fits easily into existing algorithms, making it a powerful yet flexible tool to augment multi-objective multiagent learning.

We now present some limitations of our approach and some future work. D_{MO} relies on computing the hypervolume to derive agent-level credit. This subjects it to the two biggest drawbacks of using the hypervolume indicator—that hypervolume is NP-hard to calculate exactly, and that this computation scales exponentially in the number of objectives. For few-objective problems, however, this indicator works well, as supported by our results. For many-objective problems, an effective approximation, or replacement for the hypervolume indicator would be necessary. Thus, as future work, we would like to explore other multi-objective indicators that can replace the hypervolume computation. One potential substitute is the nondominated rank of a solution, and to measure the effect of single agent policies on this rank with D_{MO}'s counterfactual replacements.

8 Acknowledgements

This work was partially supported by the Air Force Office of Scientific Research grant no. FA9550-19-1-0195 and National Science Foundation grant no. IIS-2112633.

References

[1] Adrian Agogino, Kagan Tumer, and Risto Miikkulainen. 2005. Efficient credit assignment through evaluation function decomposition. In *Proceedings of the 7th*

Annual Conference on Genetic and Evolutionary Computation (Washington DC, USA) *(GECCO '05)*. Association for Computing Machinery, New York, NY, USA, 1309–1316. https://doi.org/10.1145/1068009.1068221

[2] W. Brian Arthur. 1994. Inductive Reasoning and Bounded Rationality. *The American Economic Review* 84, 2 (1994), 406–411. http://www.jstor.org/stable/2117868

[3] Tim Brys, Anna Harutyunyan, Peter Vrancx, Matthew E. Taylor, Daniel Kudenko, and Ann Nowe. 2014. Multi-objectivization of reinforcement learning problems by reward shaping. In *2014 International Joint Conference on Neural Networks (IJCNN)*. 2315–2322. https://doi.org/10.1109/IJCNN.2014.6889732

[4] Jacopo Castellini, Sam Devlin, Frans A. Oliehoek, and Rahul Savani. 2021. Difference Rewards Policy Gradients. In *20th International Conference on Autonomous Agents and Multiagent Systems*. https://www.microsoft.com/en-us/research/publication/dr-reinforce/

[5] Xinning Chen, Xuan Liu, Yanwen Ba, Shigeng Zhang, Bo Ding, and Kenli Li. 2023. Selective learning for sample-efficient training in multi-agent sparse reward tasks. In *ECAI 2023*. IOS Press, 413–420.

[6] Mitchell Colby and Kagan Tumer. 2012. Shaping fitness functions for coevolving cooperative multiagent systems, Vol. 1. 425–432.

[7] Joshua Cook and Kagan Tumer. 2022. Fitness shaping for multiple teams. In *Proceedings of the Genetic and Evolutionary Computation Conference* (Boston, Massachusetts) *(GECCO '22)*. Association for Computing Machinery, New York, NY, USA, 332–340. https://doi.org/10.1145/3512290.3528829

[8] Joshua Cook, Kagan Tumer, and Tristan Scheiner. 2023. Leveraging Fitness Critics To Learn Robust Teamwork. In *Proceedings of the Genetic and Evolutionary Computation Conference* (Lisbon, Portugal) *(GECCO '23)*. Association for Computing Machinery, New York, NY, USA, 429–437. https://doi.org/10.1145/3583131.3590497

[9] David Corne, N. Jerram, Joshua Knowles, and Martin Oates. 2001. PESA-II: Region-based selection in evolutionary multiobjective optimization. *Proc. 6th Int. Conf. Pparallel Prob. Solving from Nature PPSN-VI* (01 2001).

[10] Matteo Cristani, Luca Pasetto, and Claudio Tomazzoli. 2020. Protecting the environment: a multi-agent approach to environmental monitoring. *Procedia Computer Science* 176 (2020), 3636–3644.

[11] Indraneel Das and John E. Dennis. 1997. A closer look at drawbacks of minimizing weighted sums of objectives for Pareto set generation in multicriteria optimization problems. *Structural optimization* 14 (1997), 63–69. https://api.semanticscholar.org/CorpusID:51769863

[12] Jean-Pierre de la Croix, Federico Rossi, Roland Brockers, Dustin Aguilar, Keenan Albee, Elizabeth Boroson, Abhishek Cauligi, Jeff Delaune, Robert Hewitt, Dima Kogan, Grace Lim, Benjamin Morrell, Yashwanth Nakka, Viet Nguyen, Pedro Proença, Gregg Rabideau, Joseph Russino, Maira Saboia da Silva, Guy Zohar, and Subha Comandur. 2024. Multi-Agent Autonomy for Space Exploration on the CADRE Lunar Technology Demonstration. In *2024 IEEE Aerospace Conference*. 1–14. https://doi.org/10.1109/AERO58975.2024.10521425

[13] Kalyanmoy Deb, Ram Bhushan Agrawal, et al. 1995. Simulated binary crossover for continuous search space. *Complex systems* 9, 2 (1995), 115–148.

[14] Kalyanmoy Deb, Samir Agrawal, Amrit Pratap, and T. Meyarivan. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. Evol. Comput.* 6 (2002), 182–197. https://api.semanticscholar.org/CorpusID:9914171

[15] Sam Devlin, Logan Yliniemi, Daniel Kudenko, and Kagan Tumer. 2014. Potential-based difference rewards for multiagent reinforcement learning. In *Proceedings of the 2014 International Conference on Autonomous Agents and Multi-Agent Systems* (Paris, France) *(AAMAS '14)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 165–172.

[16] Amir Ebrahimi Zade, Ahmad Sadegheih, and Mohammad Mehdi Lotfi. 2014. A modified NSGA-II solution for a new multi-objective hub maximal covering problem under uncertain shipments. *Journal of Industrial Engineering International* 10, 4 (Dec. 2014), 185–197.

[17] Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. 2018. Counterfactual multi-agent policy gradients. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 32.

[18] Andreia P. Guerreiro, Carlos M. Fonseca, and Luís Paquete. 2021. The Hypervolume Indicator: Computational Problems and Algorithms. *ACM Comput. Surv.* 54, 6, Article 119 (jul 2021), 42 pages. https://doi.org/10.1145/3453474

[19] Hodjat Hamidi and Ali Kamankesh. 2018. An approach to intelligent traffic management system using a multi-agent system. *International Journal of Intelligent Transportation Systems Research* 16 (2018), 112–124.

[20] Jeffrey Horn, N. Nafpliotis, and D.E. Goldberg. 1994. A Niched Pareto Genetic Algorithm for Multi-Objective Optimization. *Proceedings of the 1st IEEE Conference on Computation Evolutionary* 1, 82 – 87 vol.1. https://doi.org/10.1109/ICEC.1994.350037

[21] Yuzheng Hu, Ruicheng Xian, Qilong Wu, Qiuling Fan, Lang Yin, and Han Zhao. 2024. Revisiting scalarization in multi-task learning: A theoretical perspective. *Advances in Neural Information Processing Systems* 36 (2024).

[22] J. Knowles and D. Corne. 1999. The Pareto archived evolution strategy: a new baseline algorithm for Pareto multiobjective optimisation. In *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)*, Vol. 1. 98–105 Vol. 1. https://doi.org/10.1109/CEC.1999.781913

[23] Xiaopeng Li. 2022. Trade-off between safety, mobility and stability in automated vehicle following control: An analytical method. *Transportation Research Part B: Methodological* 166 (2022), 1–18. https://doi.org/10.1016/j.trb.2022.09.003

[24] Patrick Mannion, Sam Devlin, Karl Mason, Jim Duggan, and Enda Howley. 2017. Policy invariance under reward transformations for multi-objective reinforcement learning. *Neurocomputing* 263 (2017), 60–73. https://doi.org/10.1016/j.neucom.2017.05.090 Multiobjective Reinforcement Learning: Theory and Applications.

[25] Patrick Mannion, Jim Duggan, and Enda Howley. 2017. Analysing the effects of reward shaping in multi-objective stochastic games. (2017).

[26] Patrick Mannion, Karl Mason, Sam Devlin, Jim Duggan, and Enda Howley. 2016. Multi-Objective Dynamic Dispatch Optimisation using Multi-Agent Reinforcement Learning: (Extended Abstract). In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems* (Singapore, Singapore) *(AAMAS '16)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 1345–1346.

[27] Mohammad Marufuzzaman, Ridvan Gedik, and Mohammad S Roni. 2016. A Benders based rolling horizon algorithm for a dynamic facility location problem. *Computers & Industrial Engineering* 98 (2016), 462–469.

[28] Kaisa Miettinen and Marko Mäkelä. 2002. On scalarizing functions in multiobjective optimization. *OR Spectrum* 24 (01 2002), 193–213. https://doi.org/10.1007/s00291-001-0092-9

[29] Duc Thien Nguyen, Akshat Kumar, and Hoong Chuin Lau. 2018. Credit Assignment For Collective Multiagent RL With Global Rewards. In *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Eds.), Vol. 31. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2018/file/94bb077f18daa6620efa5cf6e6f178d2-Paper.pdf

[30] Anna Nickelson, Nicholas Zerbel, Gaurav Dixit, and Kagan Tumer. 2023. Shaping the Behavior Space with Counterfactual Agents in Multi-Objective Map Elites. In *Proceedings of the 15th International Joint Conference on Computational Intelligence - Volume 1: ECTA*. INSTICC, SciTePress, 41–52. https://doi.org/10.5220/0012164800003595

[31] Gennaro Notomista, Claudio Pacchierotti, and Paolo Robuffo Giordano. 2022. Multi-Robot Persistent Environmental Monitoring Based on Constraint-Driven Execution of Learned Robot Tasks. In *2022 International Conference on Robotics and Automation (ICRA)*. 6853–6859. https://doi.org/10.1109/ICRA46639.2022.9811673

[32] Liviu Panait and Sean Luke. 2005. Cooperative Multi-Agent Learning: The State of the Art. *Autonomous Agents and Multi-Agent Systems* 11, 3 (Nov. 2005), 387–434. https://doi.org/10.1007/s10458-005-2631-2

[33] Roxana Radulescu, Patrick Mannion, Diederik M. Roijers, and Ann Nowé. 2020. Multi-Objective Multi-Agent Decision Making: A Utility-based Analysis and Survey. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems* (Auckland, New Zealand) *(AAMAS '20)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 2158–2160.

[34] Aida Rahmattalabi, Jen Jen Chung, Mitchell Colby, and Kagan Tumer. 2016. D++: Structural credit assignment in tightly coupled multiagent domains. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 4424–4429. https://doi.org/10.1109/IROS.2016.7759651

[35] Nery Riquelme, Christian Von Lücken, and Benjamin Baran. 2015. Performance metrics in multi-objective optimization. In *2015 Latin American Computing Conference (CLEI)*. 1–11. https://doi.org/10.1109/CLEI.2015.7360024

[36] Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever. 2017. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv preprint arXiv:1703.03864* (2017).

[37] Felipe Petroski Such, Vashisht Madhavan, Edoardo Conti, Joel Lehman, Kenneth O Stanley, and Jeff Clune. 2017. Deep neuroevolution: Genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning. *arXiv preprint arXiv:1712.06567* (2017).

[38] Kagan Tumer and David H Wolpert. 2000. Collective intelligence and Braess' paradox. In *Aaai/iaai*. 104–109.

[39] Peter Vamplew, Benjamin J. Smith, Johan Källström, Gabriel Ramos, Roxana Rădulescu, Diederik M. Roijers, Conor F. Hayes, Fredrik Heintz, Patrick Mannion, Pieter J. K. Libin, Richard Dazeley, and Cameron Foale. 2022. Scalar reward is not enough: a response to Silver, Singh, Precup and Sutton (2021). *Autonomous Agents and Multi-Agent Systems* 36, 2 (Oct. 2022), 19 pages. https://doi.org/10.1007/s10458-022-09575-5

[40] Peter Vamplew, John Yearwood, Richard Dazeley, and Adam Berry. 2008. On the Limitations of Scalarisation for Multi-objective Reinforcement Learning of Pareto Fronts. In *Proceedings of the 21st Australasian Joint Conference on Artificial Intelligence: Advances in Artificial Intelligence* (Auckland, New Zealand) *(AI '08)*. Springer-Verlag, Berlin, Heidelberg, 372–378. https://doi.org/10.1007/978-3-540-89378-3_37

[41] Marios Xanthidis, Bharat Joshi, Jason M. O'Kane, and Ioannis Rekleitis. 2022. Multi-Robot Exploration of Underwater Structures. *IFAC-PapersOnLine* 55, 31

(2022), 395–400. https://doi.org/10.1016/j.ifacol.2022.10.460 14th IFAC Conference on Control Applications in Marine Systems, Robotics, and Vehicles CAMS 2022.

[42] Logan Yliniemi and Kagan Tumer. 2014. Multi-objective Multiagent Credit Assignment Through Difference Rewards in Reinforcement Learning. In *Simulated Evolution and Learning*, Grant Dick, Will N. Browne, Peter Whigham, Mengjie Zhang, Lam Thu Bui, Hisao Ishibuchi, Yaochu Jin, Xiaodong Li, Yuhui Shi, Pramod Singh, Kay Chen Tan, and Ke Tang (Eds.). Springer International Publishing, Cham, 407–418.

[43] Logan Yliniemi and Kagan Tumer. 2016. Multi-objective multiagent credit assignment in reinforcement learning and NSGA-II. *Soft Computing* 20, 10 (01 Oct 2016), 3869–3887. https://doi.org/10.1007/s00500-016-2124-z

[44] Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele. 2000. Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation* 8, 2 (2000), 173–195. https://doi.org/10.1162/106365600568202

[45] Eckart Zitzler, Marco Laumanns, and Lothar Thiele. 2001. SPEA2: Improving the strength pareto evolutionary algorithm. https://api.semanticscholar.org/CorpusID:16584254