VIETNAM NATIONAL UNIVERSITY HANOI
UNIVERSITY OF ENGINEERING AND TECHNOLOGY

LE BANG GIANG

APPLY GRAPH NEURAL NETWORK FOR SEARCHING PARETO OPTIMALITY IN MULTI-AGENT REINFORCEMENT LEARNING

Field: Computer Science
Major: Computer Science
Code: 8480101.01

SUMMARIZTION OF COMPUTER SCIENCE
MASTER'S THESIS

Hanoi - 2024

Chapter 1

Introduction

1.1. Motivation

Multi-Agent Reinforcement Learning (MARL) is a framework where multiple agents learn optimal decision-making through reinforcement learning. While MARL has achieved remarkable results in zero-sum games and fully cooperative environments, many real-world problems involve self-interested agents with potentially conflicting objectives. These settings require balancing cooperation and competition to achieve Pareto optimality, where no policy is strictly better than another. Current MARL methods often converge to Nash equilibria, which may be suboptimal for such problems. To address this, the thesis introduces altruistic learning, where agents optimize not only their rewards but also those of others, linking MARL with multi-objective optimization (MOO). The Multiple Gradient Descent Algorithm (MGDA), a popular MOO method, struggles with weak Pareto solutions in MARL, prompting the development of MGDA++, an improved algorithm shown to achieve strong Pareto optimality. The work also introduces a novel sufficient condition for strong Pareto optimality with convex objectives. Furthermore, challenges such as partial observability, non-stationarity, and miscoordination are tackled by incorporating graph neural networks (GNNs), which enhance information aggregation among agents, leading to better performance in cooperative settings.

1.2. Contributions

The main contributions of the thesis are

- **Connect MOO with MARL**.
- **Novel MOO algorithm for finding Strong Pareto solutions**.
- **Apply GNN to handle agents' interactions**.
- **Empirical demonstration on two benchmarks**.

1.3. Outlines

This thesis is structured as follows:

- Chapter 1: Introduction.
- Chapter 2: Background and Related works.
- Chapter 3: A new method for finding Strong Pareto policies.
- Chapter 4: Experiments.
- Chapter 5: Conclusion.

Chapter 2

Backgrounds and Related Works

2.1. Multi-Agent Reinforcement Learning

Markov game. Multi-Agent Reinforcement Learning is modeled as a Markov game, defined by $\langle \mathcal{N}, \mathcal{S}, \mathcal{A}, P, r, \gamma, \rho \rangle$, where $\mathcal{N}$ is the set of N agents, $\mathcal{S}$ the state space, and $\mathcal{A} = (\mathcal{A}_1 \times \cdots \times \mathcal{A}N)$ the joint action space. ρ is the initial state distribution, and $\gamma \in [0, 1)$ the discount factor. Each agent i has a reward function $r^i : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$. The expected returns are defined as $J_i(\pi) = \mathbb{E}_\pi \sum_t \gamma^t r_t^i$. Unlike cooperative settings, this work focuses on finding Pareto optimal solutions, defined later.

Optimality concepts in MARL. Key concepts include Pareto optimality (PO), where no agent improves without hurting others, and Nash Equilibrium (NE), where no agent has incentive to change their strategy.

Fully-cooperative vs. cooperative settings. Fully cooperative settings use reward aggregation, sharing a single group reward, which leads to credit assignment challenges. In cooperative settings, agents independently optimize their rewards, often via independent learning (e.g., IQL, IPPO, MADDPG). Both settings face issues like suboptimal NE convergence.

MAPPO. MAPPO adapts PPO for fully cooperative settings, using a centralized

policy π and value function V. It employs a trust region clipping mechanism:

$$L^{\text{MAPPO}}_{\pi_{\theta_{\text{old}}}}(\pi_\theta) = \sum_i^n \mathbb{E}_{s,\mathbf{a}\sim\pi_{\theta_{\text{old}}}}\left[\min\left(\frac{\pi_\theta(\mathbf{a}^i|s)}{\pi_{\theta_{\text{old}}}(\mathbf{a}^i|s)}A_{\pi_{\theta_{\text{old}}}}(s,\mathbf{a}), \text{clip}\left(\frac{\pi_\theta(\mathbf{a}^i|s)}{\pi_{\theta_{\text{old}}}(\mathbf{a}^i|s)}, 1\pm\varepsilon\right)A_{\pi_{\theta_{\text{old}}}}(s,\mathbf{a})\right)\right] \tag{2.1}$$

where $A_{\pi_{\theta_{\text{old}}}}$ is the advantage function.

IQL. Independent Q-Learning extends DQN to multi-agent domains, optimizing each agent's Q-function:

$$L_{\text{TD}} = \mathbb{E}\left[\left(r+\gamma \text{sg}\left(\max_{a'} Q_{\theta'}(s',a')\right) - Q_\theta(s,a)\right)^2\right], \tag{2.2}$$

where $\text{sg}(\cdot)$ stops gradients, and θ' are the target network parameters.

IPPO. IPPO decentralizes PPO for local observations, achieving decentralized training and execution. While theoretical concerns exist, centralized trust regions alleviate instability.

2.2. Multi-Objective Optimization

Notations. Define $[n] = 1,\ldots,n$, norms $|\cdot|$ as L2 norms, and $\langle\cdot,\cdot\rangle$ as dot products. For vectors x, y, x^i is the i^{th} element, with $x \preceq y$ and $x \prec y$ indicating elementwise inequality.

Problem setting and definitions. Consider minimizing $F : \mathbb{R}^m \to \mathbb{R}^n$:

$$\min_{x\in\mathbb{R}^m} F(x) = [F_1(x), F_2(x), \ldots, F_n(x)]. \tag{2.3}$$

Steepest gradient descent. This method finds an update vector d minimizing:

$$\min_d \max_{i=1,\ldots,n} \langle \nabla F_i(x), d\rangle + \frac{1}{2}\|d\|^2. \tag{2.4}$$

Reformulated as:

$$\begin{aligned} &\text{minimize} && \alpha + \frac{1}{2}\|d\|^2 \\ &\text{s.t.} && \langle \nabla F_i(x), d\rangle - \alpha \le 0, \quad \forall i = 1,\ldots,n. \end{aligned} \tag{2.5}$$

MGDA. The dual formulation is:

$$\begin{array}{ll} \text{minimize} & \|\sum_i^n \lambda_i \nabla F_i(x)\|^2 \\ \text{s.t.} & \sum_i \lambda_i = 1 \\ & \lambda_i \geq 0 \qquad \forall i = 1, \ldots, n. \end{array} \tag{2.6}$$

2.3. Graph Neural Network for MARL

Agents in MARL can be modeled as nodes in a dynamic graph with neighbors B_i.

DGN. DGN uses convolutional GNNs with attention-based kernels:

$$\alpha_{ij}^m = \frac{\exp(\tau \langle W_Q^m h_i, W_K^m h_j \rangle)}{\sum_k^{B_i} \exp(\tau \langle W_Q^m h_i, W_K^m h_k \rangle)}, \tag{2.7}$$

updating latent vectors:

$$h^{i'} = \text{MLP}\left(\text{concat}m \left[\sum j \in B_i \alpha_{ij}^m W_V^m h_j\right]\right). \tag{2.8}$$

2.4. Related Works

Multi-Agent Reinforcement Learning (MARL) for Pareto Optimal Policies. In fully cooperative MARL, Pareto optimality simplifies to a single global value function, aligning weak and strong Pareto solutions. Key approaches include value decomposition and policy-gradient methods. Early research explored independent learning for agents, which can converge to Nash equilibria. Centralized training frameworks like those of Lowe et al. improved cooperative-competitive tasks. Recent studies (e.g., Zhao et al.) highlight suboptimality in MARL methods and motivate Pareto-optimal policy exploration. Christianos et al. introduced Pareto Actor-Critic, but its computational cost and reliance on fully cooperative environments limit its applicability.

Multi-Objective Optimization (MOO)/Multitask Learning. Gradient-based MOO approaches, such as MGDA, have been adapted to find Pareto stationary points, with enhancements for high-dimensional gradients. Most works treat weak and strong Pareto optimality equivalently, though some (e.g., Roy et al.) explore conditions where they coincide. Linearized approaches and conflict-mitigating methods (e.g., Yu et al.) balance objectives but fail to explore diverse

Pareto solutions fully. The proposed thesis algorithm builds on MGDA to enhance strong Pareto convergence while retaining Pareto front exploration.

Graph Modeling in MARL. Graph Neural Networks (GNNs) are leveraged to model agent communication and interaction, addressing challenges like non-stationarity and partial observability. Recent works, such as CommFormer and hierarchical GNNs, have advanced information sharing and coordination among agents. Despite their potential, these methods are often tailored to specific tasks, requiring prior knowledge of the environment. To reduce complexity, this thesis focuses on Convolutional Neural Networks (CNNs) instead of GNNs for cooperative MARL, avoiding premature dependence on graph-based methods.

Chapter 3

Method

3.1. Challenges in Achieving Pareto Optimality in MARL Frameworks

A persistent challenge in cooperative MARL is that PG algorithms often converge to Pareto-dominated policies. While PG methods can find Nash Equilibria (NE) in Markov Potential Games, they fail to achieve Pareto optimality even with arbitrary communication between agents. For example, in the matrix game in Figure 3.1, any policy profile is a NE, but only (A,A) is Pareto optimal. Here, PG fails as the gradients are zero when the agent's reward does not depend on its actions. This limitation also extends to value-based methods like IQL and MADDPG, where selfish optimization leads to Pareto-dominated outcomes. To address this, agents must transition from individual reward optimization to considering others' objectives during training.

		Player 2	
		A	B
Player 1	A	$1,2$	$0,2$
	B	$1,0$	$0,0$

Figure 3.1: Example of the matrix game, the tuple in each table cell contains the reward of agent 1 and 2, respectively.

3.2. MGDA with Trust Region Policy Optimization

To enable Pareto optimality in MARL, I integrate MAPPO with MGDA. The policy objective for agent i concerning reward j is modified as:

$$L_{\text{PPO}}^{i,j} = \min\left(\frac{\pi_{\text{new}}^{i}}{\pi_{\text{old}}^{i}}\hat{A}_{j}^{\pi_{\text{old}}^{i}}, \text{clip}\left(\frac{\pi_{\text{new}}^{i}}{\pi_{\text{old}}^{i}}, 1-\varepsilon, 1+\varepsilon\right)\hat{A}_{j}^{\pi_{\text{old}}^{i}}\right), \tag{3.1}$$

where $\hat{A}_{j}^{\pi_{\text{old}}^{i}}$ is the estimated advantage for reward j, with baselines from a multi-head value function. Gradients from all agents' rewards are passed into MGDA (or MGDA++) to find a descent direction, ensuring improvement across all objectives.

3.3. The problem of MGDA

MGDA often converges to weak Pareto optimal points, which may be suboptimal compared to strong Pareto solutions. In convex settings, any Pareto stationary point is weakly optimal. This limitation is exacerbated by the influence of diminishing gradients from converged objectives, as illustrated by a bi-objective example in Figure 3.3. While gradient normalization can alleviate slow convergence empirically, it lacks theoretical justification. To address this, I propose MGDA++, which rectifies MGDA's weaknesses by ensuring Strong Pareto Convergence, even in complex MARL settings. The pseudocode is presented in 1.

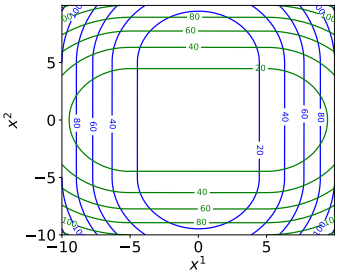


Figure 3.2: Objective landscape in 2D space.

We give the theoretical results of the proposed algorithm.

Assumption 1. *All objective functions F_i are convex and L-smooth.*

Assumption 2. *With a given point x_0, the sum of objectives function F has a bounded level set $\Gamma = \{x | \sum_i F_i(x) \leq \sum_i F_i(x_0)\}$. Furthermore, the optimal set of each objective function F_i is non-empty and is denoted by X_i^*, the optimal value is denoted similarly as F_i^*.*

Algorithm 1 MGDA++ Algorithm

Input: $\varepsilon > 0$, initial solution x_0

for $k = 0, 1, \ldots$ **do**
  $S_k \leftarrow \emptyset$
  for $i = 1, \ldots, n$ **do**
    Calculate $\nabla F_i(x_k)$
    if $\|\nabla F_i(x_k)\| > \varepsilon$ **then**
      $S_k = S_k \cup \{i\}$
    end if
  end for
  if $S_k = \emptyset$ **then**
    Stop
  end if
  Find $\{\lambda_i\}_{i \in S_k}$ by solving (2.6) on the subset of gradients $\{\nabla F_i(x_k)\}_{i \in S_k}$
  $d_k \leftarrow \sum_{i \in S_k} \lambda_i \nabla F_i(x_k)$
  Choose step size t_k
  $x_{k+1} \leftarrow x_k - t_k d_k$
end for

Lemma 1. *Under assumption 1, if there exists a convex combination of the subset of all non-zero gradient vectors*

$$\sum_{i \in S} \lambda_i \nabla F_i(x) = 0; \quad \lambda_i > 0, \quad \|\nabla F_i(x)\| > 0 \quad \forall i \in S \tag{3.2}$$

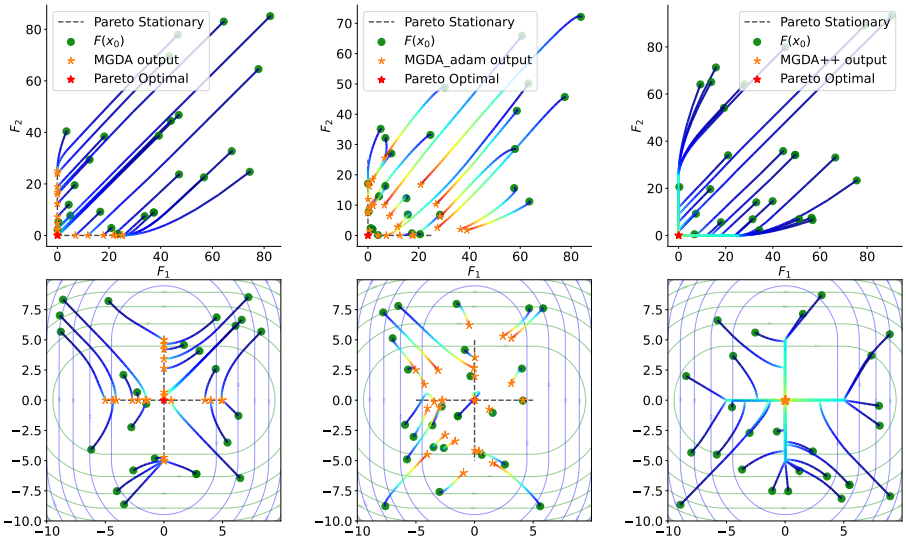
with $S \subseteq [n], S \neq \emptyset$, then x is Pareto optimal.

Proposition 2. *Under assumptions 1 and 2, then for any $\varepsilon > 0$, it is possible to choose an $0 < \varepsilon \leq \sqrt{2L\varepsilon}$ such that if any x satisfies $\|\nabla F_i(x)\| < \varepsilon$, then $F_i(x) \leq F_i^* + \varepsilon, \forall i \in [n]$. Such x are necessarily ε-Pareto optimal solutions.*

Theorem 3. *Under assumptions 1 and 2, for any $\varepsilon > 0$, with $n = 2$ and by choosing $\varepsilon < \sqrt{2L\varepsilon}$ and with appropriate choices of each update steps t_k as*

$$t_k = \begin{cases} \max\left(\dfrac{|S_k| \|d_k\|^2 + \left\langle \sum_{i \in \overline{S}_k} \nabla F_i(x_k), d_k \right\rangle}{nL\|d_k\|^2}, 0 \right) & \text{if } \|d_k\| > 0 \\ 0 & \text{if } \|d_k\| = 0 \end{cases}, \tag{3.3}$$

with $\overline{S}_k$ the complement of S_k, then each convergent subsequence of MGDA++ converges to either Pareto optimal or ε-Pareto optimal solutions.



(a) MGDA (b) MGDA w/ Adam (c) MGDA++

Figure 3.3: Comparison of MGDA, MGDA with Adam and MGDA++. Left: MGDA gets stuck at Pareto Stationary points, where the learning completely stops. Middle: Changing the optimizer does not help in avoiding the suboptimal convergence. Right: MGDA++ is able to converge to strong Pareto Optimal solutions while avoiding being trapped at Pareto Stationary points.

3.4. Applying Graph Neural Networks for Handling Agent Interactions

I next examine the application of Convolutional Graph Neural Networks within my algorithm in Section 3.2.. Specifically, I base my implementation of GCN for MARL on DGN (Jiang et al. 2018). The observations of each agent are first encoded by a feature encoder f_e and then fed into the convolutional layer, which is a multi-head attention network.

With each attention head, I project the feature encoding of the previous layer into query, key, and value vectors. The value feature vector is then the weighted sum of the value vectors, whose weights are calculated as a softmax distribution of the dot product of the corresponding key and query vectors between neighboring agents in the adjacency graph. The output of a convolutional layer is a concatenation of all the attended value vectors, projected into another

space with a smaller dimension using a single-layer MLP.

Several design choices, such as network architecture, are deferred to the Hyperparameters section. The pseudocode of the Graph Neural Network in our method is detailed in Algorithm 2.

Here, I highlight the main differences between my Convolutional Graph approach and DGN: 1) DGN follows the single-agent optimization approach discussed in Section 3.1., while our method follows the MOO optimization framework with altruistic learning, and 2) DGN employs a value-based, off-policy RL algorithm (DQN), while I base my algorithm on the on-policy, trust region method MAPPO (Yu et al., Section 3.2.).

Algorithm 2 Graph Neural Policy for multi-agent RL

Input: Observations $\{o_i\}_i^N$ of all agents, number of agents N, encoder network f_e, weight matrices of query, key and value $(W_{Qk}^m, W_{Kk}^m, W_{Vk}^m)$, number of Graph Convolutional layer K (i.e hops), number of multi-head attention M, Adjacency matrix B

Output: Actions $\{a_i\}_i^N$

Encode the observations by the encoder to get latent $z_i = f_e(o_i), \forall i \leq N$
Let $h_0^i := z_i, \forall i \leq N$
for $k = 0, \ldots, K-1$ **do**
    for $i = 1, \ldots, N$ **do**
        Calculate attention weights $\{\alpha_{ij}^m\}_j^{B_i}$ according to Eq (2.7) $\forall m \leq M$
        Calculate the value vectors $V_i^m = \sum_j^{B_i} a_{ij}^m W_{Vk}^m h_k^j, \quad \forall m \leq M$
        Calculate $h_{k+1}^i = MLP(\text{concat}[V_i^m, \forall m \leq M])$ (Eq 2.8).
    end for
end for
for $i = 1, \ldots, N$ **do**
    // Find logit (discrete) or mean and std (gaussian)
    $h^i = MLP(\text{concat}[h_k^i, k \leq K])$
    // Sample from distribution (e.g. Categorical, Gaussian, or Squashed Gaussian)
    $a_i \sim \text{dist}(h_i)$
end for

Chapter 4

Experiments

In this chapter, I present the experimental setup for evaluating the proposed method against baselines, addressing three questions: 1) Does altruistic learning help find Pareto solutions? 2) Does MGDA++ improve strong Pareto optimality compared to baselines? 3) How does incorporating graph neural networks impact overall performance?

4.1. Experimental Setup

4.1.1. Benchmarks

The methods are evaluated on two MARL benchmarks: Gridworld and MPE.

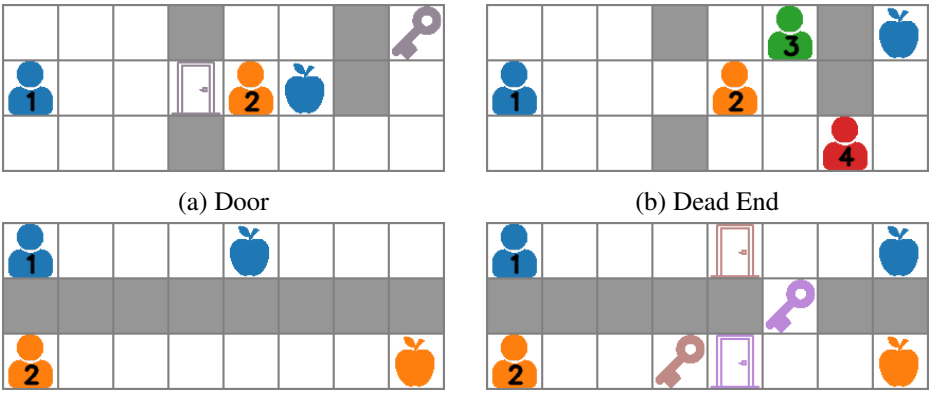
Gridworld. Gridworld environments are inspired by altruistic learning setups (Franzmeyer et al., 2022), where agents navigate grids, earning rewards ($+10$) for reaching color-matched goals and incurring penalties (-0.1) for collisions. Doors, opened by agents occupying matching-color key positions, add a cooperation element. Four variations of the environment are studied (Figure 4.1):

a) **Door** (Franzmeyer et al., 2022): The second agent opens a door for the first, enabling altruistic behaviors to succeed, as selfish learners fail.

b) **Dead End** (Franzmeyer et al., 2022): Complex MARL dynamics emerge as agents block paths, illustrating interference in optimization problems.

c) **Two Corridors**: Independent agents face asymmetric task difficulty, leading to weak Pareto solutions due to varying convergence rates.

d) **Two Rooms**: Implicit cooperation arises as one agent can open the other's door, balancing altruism and self-interest without explicit incentives.



(a) Door (b) Dead End

(c) Two Corridors (d) Two Rooms

Figure 4.1: Four scenarios of the Gridworld environment.

Multiple Particle Environment (MPE) (Lowe et al., 2017). MPE consists of scenarios with various reward structures, ranging from competitive to cooperative. Agents are represented as particles in a 2D plane that can move, communicate, and interact. For evaluation, I focus on cooperative environments (Yu et al., 2022; Zhong et al., 2024), specifically Spread and Reference, omitting Speaker due to its heterogeneous-agent focus (Zhong et al., 2024).

1. **Reference**. Two agents and three colored landmarks. Each agent must reach its target landmark, which is only known to the other agent, necessitating communication. Both agents can speak and listen simultaneously.
2. **Spread**. Three agents and three landmarks. Agents aim to cover all landmarks while avoiding collisions, with rewards based on the distance to the closest agent.

4.1.2. Baselines

To assess my methods, I perform two experiments: (1) evaluating MGDA++ in MARL with MAPPO (Yu et al., 2022), IPPO (de Witt et al., 2020), and IQL

(Tampuu et al., 2017) as baselines, and (2) integrating a graph neural network into the framework, comparing against DGN (Jiang et al., 2018) and graph-augmented baselines.

1. **Multi-head MAPPO**. An extension of MAPPO (Yu et al., 2022) with multi-head value networks, predicting returns for each agent while following CTDE.
2. **IPPO** (de Witt et al., 2020). Decentralized PPO, where agents learn independently, ignoring non-stationarity caused by others, but performing competitively with centralized methods (Sun et al., 2022).
3. **IQL** (Tampuu et al., 2017). An off-policy, value-based extension of DQN for multi-agent settings, using independent Q-networks and replay buffers for each agent.
4. **MGPO**. Uses MAPPO's multi-head architecture but applies MGDA for a shared optimization direction across agents.
5. **MGPO++**. Similar to MGPO but employs the proposed MGDA++ method for optimization.

For graph-based methods, I extend MAPPO, MGPO, and MGPO++ with a Graph Convolutional Network (GCN) based on DGN (Jiang et al., 2018), allowing agents to aggregate features from neighbors via multi-head attention.

1. **DGN** (Jiang et al., 2018). Combines DQN (Mnih et al., 2013) with a graph neural network, where nodes represent agents and edges represent local connections. Multi-head attention aggregates neighbor features.
2. **GCN-MAPPO**. Extends MAPPO with GCN for neighbor-aware feature aggregation, optimizing individual rewards.
3. **GCN-MGPO**. Similar to GCN-MAPPO but optimizes shared objectives via MGDA.
4. **GCN-MGPO++**. Similar to GCN-MGPO but uses MGDA++ for optimization.

4.1.3. Hyperparameter settings

Default hyperparameters from the original papers are used for all baselines. Three-layer MLPs with hidden sizes of 64 are employed for all methods. MAPPO,

IPPO, and IQL follow their repository defaults; MGPO and MGPO++ adopt MAPPO's hyperparameters. In MGDA++, ε is set to 0.05, except in Dead End where it is 0.1. Methods are trained for 500k steps in Gridworld and 5M steps in MPE, without parameter sharing in actor or value networks for fair comparison.

Graph-based methods use GCN with the architecture from Jiang et al. (2018). Dense communication graphs are assumed due to a small number of agents. To reduce computation, graph methods use one convolutional layer and one attention head. The encoder is a two-layer MLP for MPE and a three-layer CNN for Gridworld, while other MLPs are single-layer with ReLU activations.

4.2. Results

4.2.1. Without Graph Neural Network

In the Gridworld environment, our method outperforms all baselines across various scenarios.

In the Door scenario, MAPPO with multi-objective optimization is the only method that converges to the optimal solution for the first agent, while all other algorithms with single objectives fail to learn cooperation.

In the Dead End scenario, MGPO fails to learn except for the first agent, highlighting that MGDA can only learn the optimal policy for one agent. In MGPO++, all agents can learn good solutions without being hindered once some agents converge.

In the Two Corridors scenario, all single-objective methods converge, but MGPO fails to converge for the second agent, which has the harder task.

In the Two Rooms scenario, only MGDA++ reaches the optimal policies for both agents. While MGDA can find high rewards for the first agent, the second agent is trapped after the first agent's convergence. MAPPO with an entropy coefficient of 0 allows the first agent to find high rewards initially, but this is only an artifact of the second agent's exploration. As the entropy level of both agents decreases over time, the second agent's performance drops. IQL's performance is unstable, possibly due to the non-stationarity perceived by the first agent due to the second agent's exploration.

These findings demonstrate the effectiveness of our method in achieving optimal policies in multi-agent environments.

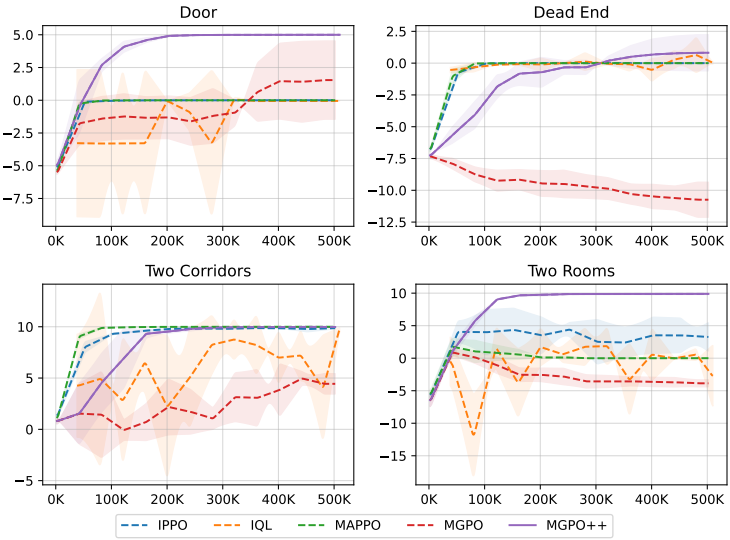


Figure 4.2: Average reward of all agents in the four scenarios. Note that, the averaging operator can be viewed as a particular linearization of the rewards vector.

MPE. In the MPE environment (Figure 4.3), the results show the following:

In the Reference scenario, MAPPO struggles as agents need to communicate to reach their goals, but are not rewarded for communication. Initially, agents find target landmarks by chance, but their performance declines as the policy becomes more deterministic. MGPO shows that the second agent freezes after the first converges, which negatively impacts the first agent's learning. In contrast, MGPO++ sees a slight increase in rewards for both agents around 3.5M timesteps, indicating improved cooperation between agents. This reward increase is absent in MGPO.

In the Spread scenario, there is little difference between methods, as the reward landscape is highly cooperative. In this setting, both strong and weak Pareto solutions align, and all Policy Gradient-based methods converge at the same rate.

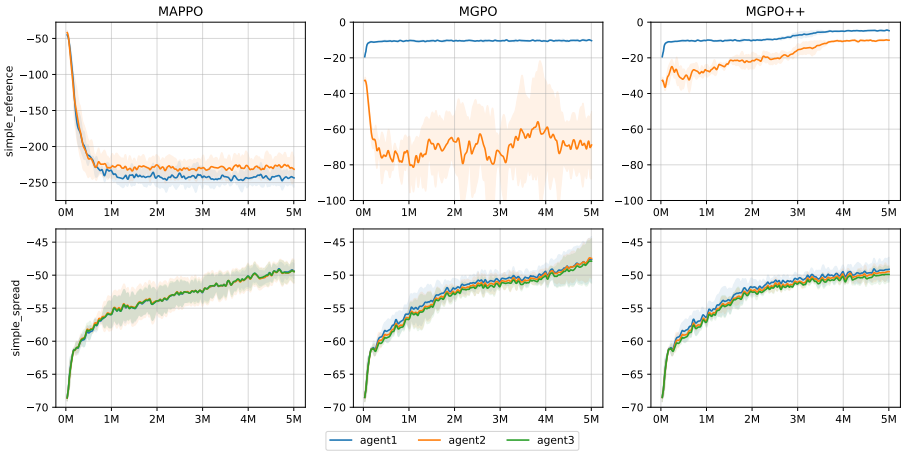


Figure 4.3: Results in MPE benchmark, showing the performance of each agent over the training period.

4.2.2. With Graph Neural Network

In this section, we present experimental results on MPE and Gridworld with graph-based methods.

Gridworld: In the Door scenario, altruistic learning methods succeed in fostering cooperation. GCN-MGPO performs similarly to the non-graph setting, with only the first agent converging, while GCN-MGPO++ converges second. Dead End is a difficult exploration problem, where most methods fail to learn good policies. GCN-MGPO shows some convergence for the first agent, but not for others.

In Two Corridors, all methods converge except for the second agent in GCN-MGPO, which suffers from weak Pareto convergence. DGN performs poorly in this simple, non-cooperative environment, suggesting it struggles with graph-based communication. GCN-MGPO++ remains superior in Two-Rooms, with both agents learning to cooperate. GCN-MGPO performs well for the first agent, but hinders the second, while GCN-MAPPO and DGN fail in this scenario.

MPE: In MPE, results are similar to the previous section. In the Spread scenario, Policy Gradient methods perform similarly, while DGN lags behind. In Reference, graph neural networks improve non-altruistic methods like GCN-MAPPO, which outperforms non-graph-based MAPPO. However, GCN-MGPO

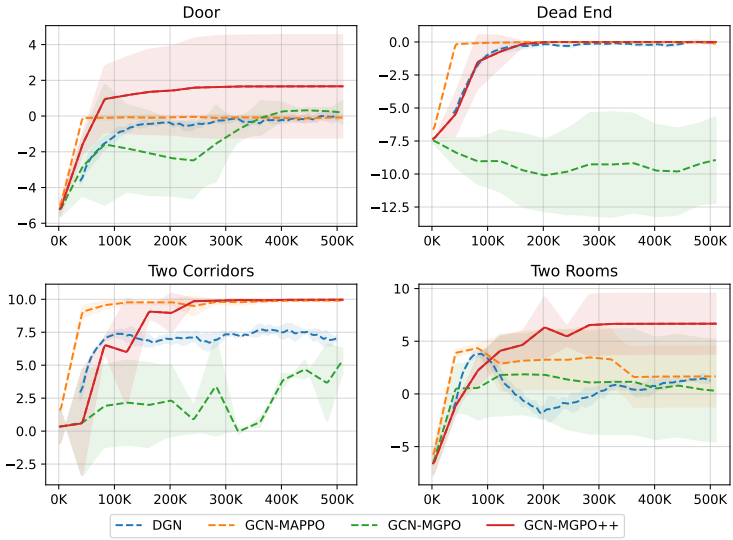


Figure 4.4: Average reward of all agents in the four scenarios with graph-based methods.

and MGPO++ show negligible improvement, as they address communication through optimization rather than information sharing.

4.2.3. Summary of Experimental Findings

MGDA++ outperforms all baselines in both Gridworld and MPE, avoiding sub-optimal policies and fostering cooperation. Graph neural networks offer limited benefits, with GCN-MAPPO showing the most improvement in Reference, but GCN-MGPO++ remains superior overall. Weak Pareto convergence persists in MGPO. Future work may explore communication pruning and other strategies to mitigate communication drawbacks in MARL.

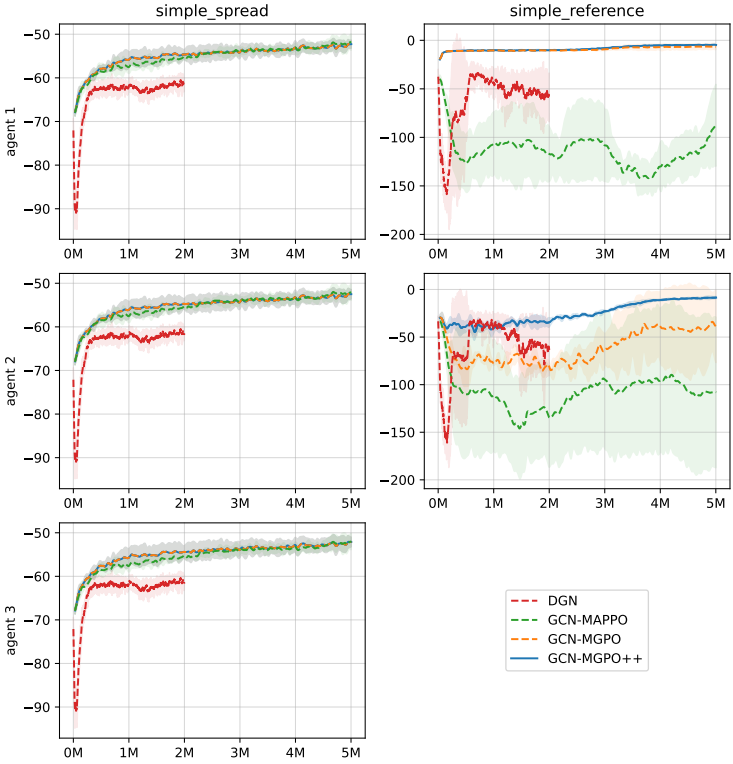


Figure 4.5: Result on MPE with graph-based methods.

Conclusion

In this thesis, I explore training multi-agents in cooperative scenarios by treating vector-based rewards as a multi-objective task, applying the MGDA algorithm to MARL. However, standard MGDA only converges to weak Pareto optimality. I introduce MGDA++ to improve convergence by filtering small norm objective gradients and provide theoretical analysis for bi-objective problems. I also apply graph neural networks (GNN) to model agent interactions. The combination of MGDA++ and GCN is evaluated in Gridworld and MPE, showing improved performance toward strong Pareto optimality.

Future Work:

- *Generalize to $n > 2$ tasks*: Extend the theoretical analysis to multiple objectives and evaluate MGDA++ on other tasks like Multi-Task Learning and Multi-Task RL.
- *More Complex Benchmarks*: Test MGPO++ on larger, more complex MARL tasks beyond Gridworld and MPE.
- *Improve GNN Component*: Enhance GNN architectures for scenarios with limited communication relevance and benchmark communication-focused MARL tasks.

Publications related to this thesis

1. Le, Bang Giang, and Viet Cuong Ta. "Toward Finding Strong Pareto Optimal Policies in Multi-Agent Reinforcement Learning." arXiv preprint arXiv:2410.19372 (2024). (To be presented at ACML 2024 Journal Track)

2. Le, Bang-Giang, Thi-Linh Hoang, Hai-Dang Kieu, and Viet-Cuong Ta. "Structural and Compact Latent Representation Learning on Sparse Reward Environments." In Asian Conference on Intelligent Information and Database Systems, pp. 40-51. Singapore: Springer Nature Singapore, 2023.

3. Le, Bang Giang, and Viet Cuong Ta. "Distill Knowledge in Multi-task Reinforcement Learning with Optimal-Transport Regularization." In 2022 14th International Conference on Knowledge and Systems Engineering (KSE), pp. 1-6. IEEE, 2022.

4. Le, Bang Giang, and Viet Cuong Ta. "On the Effectiveness of Regularization Methods for Soft Actor-Critic in Discrete-Action Domains" (Accepted with minor revision to IEEE Transactions on Systems, Man and Cybernetics: Systems).

5. Le, Bang Giang, and Viet Cuong Ta. "Low Variance Trust Region Optimization with Independent Actors and Sequential Updates in Cooperative Multi-agent Reinforcement Learning" (Major revision, submited to Journal of Autonomous Agents and Multi-Agent Systems).