

Coordination Graphs for Constrained Multi-Agent Reinforcement Learning

Santiago Amaya-Corredor, Miguel Calvo-Fullana, Anders Jonsson

Keywords: multi-agent reinforcement learning, constrained MDP, coordination graphs, factor graphs, Max-Sum, Lagrangian methods

Summary

Prior work on cooperative MARL either exploits coordination graph structure for scalability (DCG, CASEC, DMCG) without handling safety constraints, or addresses constraints via centralized training (MACPO, MAPPO-Lagrangian, CUTMAP) without exploiting factored structure. CG-CMARL bridges this gap by integrating Lagrangian constraint handling into the coordination graph framework, yielding a fully decentralized architecture in which (i) pairwise Q-functions are learned on the factor graph via parameter sharing, (ii) Max-Sum message passing coordinates actions without access to the joint action space, and (iii) a Lagrangian multiplier controls the objective–constraint tradeoff. Two shared Q-functions with separate primary and constraint output heads decouple constraint handling from learning, so that sweeping the Lagrange multiplier at evaluation time traces a Pareto front from a single trained model without retraining. We provide convergence guarantees under independent transitions and reward factorization, together with a compositional four-source error bound that reveals when factored methods incur no loss relative to centralized learning. Experiments on cooperative navigation tasks show that the resulting Pareto front dominates IQL, QMIX, DCG, MAPPO, and MAPPO-Lagrangian baselines trained at fixed reward-shaping ratios.

Contribution(s)

1. A constrained MARL framework that integrates Lagrangian constraint handling into the coordination graph structure, addressing challenges absent in single-agent constrained RL: constraint credit assignment across overlapping pairwise regions, coherent composition of primary and cost Q-functions under factored action selection, and coordinated constraint satisfaction via Max-Sum message passing. A two-head Q-function pair (primary and constraint) decouples objective and constraint learning, enabling Pareto front evaluation from a single trained model by sweeping the Lagrange multiplier at evaluation time without retraining. Parameter sharing yields $O(1)$ networks regardless of team size N , and Max-Sum has cost polynomial in N , compared to exponential for centralized approaches.
Context: Demonstrated on cooperative navigation with up to 10 agents on a pairwise graph ($O(N^2)$ regions); sparser topologies are supported by the same framework.
2. Convergence guarantees (Q-learning for both heads and primal-dual) under independent transitions and reward factorization, together with a compositional error bound decomposing into structural, coordination, sampling, and (with function approximation) representation terms. The bound provides qualitative guidance for system design by identifying when each error source vanishes or dominates.
Context: Convergence is proved in the tabular setting; with function approximation it holds up to a KKT-stationary neighborhood (Corollary 4.3).
3. Empirical validation on cooperative navigation with $N \in \{3, 4, 6, 10\}$ agents showing that the Pareto front from a single CG-CMARL model dominates IQL, QMIX, DCG, MAPPO, and MAPPO-Lagrangian baselines, each requiring separately trained models at fixed reward-shaping ratios.
Context: Empirical scope is the cooperative-navigation benchmark, which satisfies the independent-transitions assumption; extending to environments with coupled dynamics is a natural next step.

Coordination Graphs for Constrained Multi-Agent Reinforcement Learning

Santiago Amaya-Corredor¹, Miguel Calvo-Fullana¹, Anders Jonsson¹

{santiagoesteban.amaya, miguel.calvo-fullana, anders.jonsson}@upf.edu

¹Department of Engineering, Universitat Pompeu Fabra, Barcelona, Spain

Abstract

Constrained Multi-agent reinforcement learning (CMARL) faces two intertwined challenges: the joint action space grows exponentially with the number of agents, and additional requirements couple agents in ways that reward structure alone does not capture. We introduce Coordination Graphs for Constrained Multi-Agent Reinforcement Learning (CG-CMARL), a framework that addresses both challenges by combining coordination graphs with Lagrangian duality. The system decomposes the joint problem into pairwise regions, each served by a set of shared Q-functions, one for the primary objective and one for each of the constraints, so that the number of learned models is independent of the number of agents. At execution time, Max-Sum message passing coordinates actions across the factor graph, while a Lagrangian multiplier controls the objective–constraint tradeoff, allowing a single trained model to trace a Pareto front without retraining. We provide convergence guarantees under mild conditions, together with a compositional error bound that decomposes into separate interpretable sources, each traceable to a specific design choice and independently controllable. Experiments on cooperative navigation tasks (where teams of up to 10 agents must coordinate to reach target positions while satisfying pairwise constraints) show that our method produces Pareto fronts dominating established baselines trained at fixed reward-shaping ratios, while scaling to team sizes where centralized approaches become intractable.

1 Introduction

Multi-agent reinforcement learning (MARL) has emerged as a powerful framework for sequential decision-making in systems where multiple autonomous agents must coordinate their actions (Zhang et al., 2021; Albrecht et al., 2024). A central challenge is *scalability*: the joint action space grows exponentially with the number of agents, rendering centralized approaches intractable even for moderate team sizes. To address this, factored MDPs (Boutilier et al., 2000) and coordination graphs (Guestrin et al., 2002; Kok & Vlassis, 2006) exploit the locality of agent interactions, decomposing the global value function into regional components and reducing complexity from exponential in the number of agents to exponential in the size of the largest interacting group. Message-passing algorithms such as Max-Sum (Farinelli et al., 2008; Rogers et al., 2011) then enable distributed action coordination without centralized computation.

Many multi-agent tasks additionally impose *safety constraints* that cannot be traded off against reward. Collision avoidance, energy budgets, and bandwidth limits must be enforced. Constrained Markov decision processes (CMDPs) (Altman, 1999) formalize such specifications. For single-agent settings, primal-dual methods based on Lagrangian relaxation have proven effective (Tessler et al., 2018; Paternain et al., 2023), and recent work has established that CMDPs exhibit zero duality gap under mild conditions (Paternain et al., 2019).

However, extending constrained RL to the multi-agent setting introduces challenges that existing methods do not adequately address. Centralized primal-dual approaches require access to the global state and joint action space, precisely the bottleneck that factored methods aim to overcome. Independent learners cannot coordinate constraint satisfaction: if each agent optimizes its own Lagrangian independently, the team may collectively violate global constraints. The core difficulty is that *constraints couple agents* in ways that reward structure alone does not capture. To the best of our knowledge, no prior work combines the scalability of factored value functions with the constraint-handling capabilities of Lagrangian methods in a decentralized architecture with convergence guarantees.

We introduce **CG-CMARL**, a framework that addresses both scalability and constraint satisfaction. The key idea is a *two-head Q-network* shared across all pairwise regions of a coordination graph: one head predicts expected coverage reward, the other predicts expected constraint reward. A Lagrangian multiplier λ combines the two heads at action selection time, $Q_{\text{aug}} = Q_{\text{prim}} + \lambda Q_{\text{cost}}$, so that sweeping λ over a single trained model traces a Pareto front of coverage versus safety without retraining. Max-Sum message passing on the factor graph coordinates actions across agents, while parameter sharing ensures the number of learned Q-functions is $O(1)$ regardless of team size. While individual components (two-head value functions, Lagrangian multipliers, coordination graphs) exist in prior work, their integration raises challenges that do not arise in isolation: constraints couple agents across overlapping regions, the cost head must evaluate the primary-greedy policy to ensure coherent composition under factored action selection, and the compositional error bound must account for structural and coordination errors specific to the factored architecture.

- **A decentralized constrained MARL framework with Lagrangian Pareto sweep.** A two-head Q-function pair (primary and constraint) decouples objective and constraint learning, enabling a single trained model to trace the full objective–constraint Pareto front by varying λ at evaluation time without retraining. Parameter sharing across all pairwise regions yields $O(1)$ networks regardless of team size N , and damped Max-Sum message passing coordinates joint actions on the resulting factor graph with cost polynomial in N .
- **Convergence guarantees and compositional error bound.** Under independent transitions and reward factorization, we establish Q-function convergence for both heads and dual optimality (Theorems 4.1–4.2), together with a four-source error decomposition (Theorem 4.4) that identifies when each source vanishes and provides qualitative guidance for system design.
- **Empirical validation.** On the Simple Spread cooperative navigation benchmark (Lowe et al., 2017) with $N \in \{3, 4, 6, 10\}$ agents, the Pareto front from a single CG-CMARL model dominates IQL, QMIX, DCG, MAPPO, and MAPPO-Lagrangian baselines, each requiring separately trained models at fixed reward-shaping ratios.

1.1 Related Work

Our work lies at the intersection of value decomposition for cooperative MARL, constrained reinforcement learning, and distributed optimization.

Value Decomposition and Coordination Graphs. VDN (Sunehag et al., 2018) represents the joint Q-function as a sum of per-agent utilities; QMIX (Rashid et al., 2018) relaxes this to monotonic combinations via a mixing network; QTRAN (Son et al., 2019) removes monotonicity at the cost of more complex training. Coordination graphs (Guestrin et al., 2002; Kok & Vlassis, 2006) explicitly model pairwise or higher-order dependencies. Deep Coordination Graphs (DCG) (Böhmer et al., 2020) scale this to high-dimensional settings with parameter sharing, and subsequent work has explored dynamic (Siu et al., 2021), sparse (Wang et al., 2022), group-aware (Duan et al., 2024), influence-based (Zhang et al., 2025), and meta-learned topologies (Gupta et al., 2025). While these methods advance coordination graph representations, *none address constraint satisfaction*. Our work extends coordination graphs to the constrained setting by integrating Lagrangian dual variables into the factored structure.

Table 1: Comparison with related methods. “Decentr.” = no global state access at execution. “Constr.” = explicit CMDP-style constraints. “CG” = coordination graph with message passing.

Method	Decentr.	Constr.	CG
VDN / QMIX	✓	×	×
DCG / CASEC / DMCG / IESCG	✓	×	✓
MACPO / MAPPO-Lag.	×	✓	×
CUTMAP (Guan et al., 2025)	×	✓	×
Scalable CMPO (Zhang et al., 2024)	Partial	✓	×
CG-CMARL (ours)	✓	✓	✓

Constrained and Safe RL. CMDPs (Altman, 1999) formalize safety via auxiliary reward thresholds. Primal-dual methods alternate between policy improvement and multiplier updates (Borkar, 2005; Tessler et al., 2018; Stooke et al., 2020); Paternain et al. (2019) established zero duality gap, and Calvo-Fullana et al. (2024) proposed state augmentation to address limitations of standard primal-dual approaches. For multi-agent settings, MACPO (Gu et al., 2023) adapts CPO using centralized training, Zhang et al. (2024) improve scalability via truncated advantages, and CUTMAP (Guan et al., 2025) constrains cooperative policies using offline data under the CTDE paradigm. A recent survey (Kushwaha et al., 2025) identifies decentralized constraint handling in MARL as a key open problem. Our approach exploits the factored structure of both rewards and constraints, enabling decentralized learning without centralized critics.

Distributed Optimization and Credit Assignment. Subgradient methods with consensus averaging (Nedic & Ozdaglar, 2009) and two-timescale stochastic approximation (Borkar, 1997) provide the theoretical foundation for our multiplier updates. Agorio et al. (2024) applied gossiped dual variables to multi-agent assignment and considered stochastic graphs (Agorio et al., 2025). For credit assignment, difference rewards (Wolpert & Tumer, 2001) and multi-level counterfactual advantages (Zhao & Xie, 2025) decompose global feedback into agent-level signals; we adopt per-agent counterfactual rewards for regional reward shaping. In the multi-objective setting, Liu et al. (2024) discover Pareto fronts via constrained optimization steps, our Lagrangian sweep achieves a similar effect but within a factored multi-agent architecture. To our knowledge, no prior work combines coordination graph value decomposition with Lagrangian constraint handling.

Table 1 summarizes the key distinctions.

2 Problem Formulation

The goal of this section is to formalize the constrained multi-agent coordination problem and the structural assumptions that enable scalable, decentralized solutions.

Let $\mathcal{V} = \{1, \dots, N\}$ denote a team of N cooperative agents. Each agent $i \in \mathcal{V}$ has a local state space $\mathcal{S}^i$ and action space $\mathcal{A}^i$. The joint state is $s = (s^1, \dots, s^N) \in \mathcal{S} = \prod_i \mathcal{S}^i$ and the joint action is $a = (a^1, \dots, a^N) \in \mathcal{A} = \prod_i \mathcal{A}^i$. The joint action space $|\mathcal{A}| = \prod_i |\mathcal{A}^i|$ grows exponentially with N , motivating the factored approach developed below.

In many multi-agent tasks, interactions are *local*: the reward obtained by agent i depends not on all N agents but only on a small subset of neighbors. We represent these dependencies using a *coordination graph* (Guestrin et al., 2002), a hypergraph $\mathcal{G}_R = (\mathcal{V}, \mathcal{C}_R)$ where each *region* $R \in \mathcal{C}_R$ is a subset of agents whose joint state-action affects a reward component. For a region R , we write $s^R = (s^i)_{i \in R}$ and $a^R = (a^i)_{i \in R}$ for the local projections.

The coordination graph induces a factorization of the reward. We consider $J + 1$ reward signals: a primary objective r_0 and J constraint signals $r_1, \dots, r_J$, each decomposing additively over regions:

$$r_j(s, a) = \sum_{R \in \mathcal{C}_R} r_j^R(s^R, a^R), \quad j \in \{0, 1, \dots, J\}. \quad (1)$$

In our setting, $\mathcal{C}_R$ is the *pairwise* coordination graph containing all agent pairs: $\mathcal{C}_R = \{\{i, k\} : i, k \in \mathcal{V}, i \neq k\}$. This captures pairwise interactions such as collision penalties and yields $\binom{N}{2}$ regions which are quadratic in N rather than exponential.

Our approach relies on two structural assumptions:

(A1) Independent Transitions. The transition kernel factorizes across agents: $P(s' \mid s, a) = \prod_{i=1}^N P_i(s'^i \mid s^i, a^i)$. That is, each agent's next state depends only on its own current state and action. Inter-agent interactions (e.g., collisions, coordination) are captured through the reward structure (1), not through transition dynamics, a design adopted by most multi-agent environments (Lowe et al., 2017; Terry et al., 2021).

(A2) Reward Factorization. Each reward signal decomposes as in (1). We additionally assume bounded rewards: $|r_j^R(s^R, a^R)| \leq R_{\max}$ for all j, R .

Under (A1)–(A2), the value function $V_j^\pi(s) := \mathbb{E}_\pi[\sum_{t=0}^\infty \gamma^t r_j(s_t, a_t) \mid s_0 = s]$ decomposes as $V_j^\pi(s) = \sum_{R \in \mathcal{C}_R} V_{j,R}^\pi(s^R)$, where $V_{j,R}^\pi$ is the regional value function. This decomposition, which follows from linearity of expectation and the independence of transitions, is the foundation of our factored Q-learning approach. That is, the expected return of each region can be learned *independently* using only local transitions.

Remark (State vs. Observation). The theoretical analysis in Section 4 is developed for the regional Markov state s^R . In practice, each region receives an observation o^R constructed from locally available information (Section 5.1). For the pairwise coordination graph on Simple Spread, o^R contains the full Markov state of both agents in the region (positions and velocities); partial observability arises only from not observing out-of-region agents, and this discrepancy is captured by the structural error β in our compositional bound (Theorem 4.4).

With this decomposition in hand, we formulate the constrained problem. Let $c_1, \dots, c_J \in \mathbb{R}$ be constraint thresholds. We restrict attention to factorized policies $\pi(a \mid s) = \prod_i \pi^i(a^i \mid s^i)$ and seek:

$$\begin{aligned} \max_{\pi \in \Pi} \quad & V_0^\pi(s_0) \\ \text{subject to} \quad & V_j^\pi(s_0) \geq \frac{c_j}{1 - \gamma}, \quad j = 1, \dots, J. \end{aligned} \quad (2)$$

The objective maximizes expected discounted primary reward subject to each constraint reward exceeding a threshold. The normalization by $(1 - \gamma)$ converts the infinite-horizon sum to an average-reward scale, making c_j interpretable as a per-step requirement. Using the value decomposition, both objective and constraints decompose over regions, suggesting that optimization can be performed in a distributed manner.

We additionally require a regularity condition:

(A3) Feasibility (Slater's Condition). There exists a policy $\tilde{\pi} \in \Pi$ and $\xi > 0$ such that $V_j^{\tilde{\pi}}(s_0) \geq c_j/(1 - \gamma) + \xi$ for all j . This guarantees the existence of a strictly feasible policy, ensuring that the constraint set has nonempty interior and that strong duality holds for the Lagrangian relaxation.

3 Method

Our goal is to solve the constrained cooperative problem (2) in a fully decentralized manner, without access to the global state or joint action space at execution time. For clarity, we present the algorithm

for a single constraint ($J = 1$); the extension to multiple constraints requires one multiplier and one Q-function per constraint, but is otherwise straightforward. The algorithm decomposes into three components, each addressing a distinct aspect of the problem:

- *Two Q-functions per region* (Section 3.1) that separately learn primary reward and constraint reward.
- *Max-Sum message passing* (Section 3.2) for coordinated action selection on the factor graph.
- *A Lagrangian constraint mechanism* (Section 3.3) that controls the objective–constraint tradeoff, enabling Pareto front evaluation from a single trained model.

3.1 Two Q-Functions per Region

A standard approach to constrained MDPs is to form an augmented reward $\tilde{r}_\lambda = r_0 + \lambda r_1$ and learn a single Q-function for this combined signal (Tessler et al., 2018). However, this bakes the constraint tradeoff into the learned representation: changing λ requires retraining. To decouple the tradeoff from learning, we instead maintain two Q-functions per region.

For each region $R \in \mathcal{C}_R$, we define the optimal regional Q-function for the primary reward r_0^R , denoted by $Q_{\text{prim},R}(s^R, a^R)$; and the constraint reward for r_1^R under the primary Q-function’s greedy policy, $Q_{\text{cost},R}(s^R, a^R)$. Since the constraint in (2) is formulated on top of a reward instead of a cost, it is generally non-positive for cost-based constraints, and c_j is negative. Both Q-functions map regional state-action pairs to scalar values and are trained with independent TD targets. The primary Q-function performs standard Q-learning on reward r_0^R :

$$y_{\text{prim}}^R = r_0^R + \gamma \max_{a'^R} Q_{\text{prim},R}(s'^R, a'^R). \quad (3)$$

The cost Q-function performs *policy evaluation* under the primary head’s greedy policy, rather than its own maximization:

$$y_{\text{cost}}^R = r_1^R + \gamma Q_{\text{cost},R}\left(s'^R, \arg \max_{a'^R} Q_{\text{prim},R}(s'^R, a'^R)\right). \quad (4)$$

This ensures that the Q-function predicts constraint rewards under the policy induced by the primary objective, not under a cost-minimizing policy. In this sense, Q_{cost} acts as a *critic* for the primary policy with respect to the cost signal: it evaluates the consequences of primary-greedy decisions, rather than learning its own optimal cost policy. If Q_{cost} instead learned its own greedy policy via $\max_{a'^R} Q_{\text{cost},R}(s'^R, a'^R)$, it would predict the constraint reward achievable by a constraint-maximizing agent, overestimating the constraint reward that the primary-greedy agent actually achieves. Combining such a Q_{cost} with Q_{prim} in (5) would be incoherent: the two Q-functions would reflect different policies, and the Lagrangian tradeoff would be miscalibrated.

At action selection time, the two Q-functions are combined via a Lagrangian multiplier $\lambda \geq 0$:

$$Q_{\text{aug},R}(s^R, a^R) = Q_{\text{prim},R}(s^R, a^R) + \lambda Q_{\text{cost},R}(s^R, a^R). \quad (5)$$

That is, the augmented Q-function combines the primary objective with the predicted constraint reward, weighted by λ . Higher λ increases the weight on constraint satisfaction; lower λ prioritizes the primary objective. To illustrate, consider a state with two available actions:

Action	Q_{prim}	Q_{cost}	$Q_{\text{aug}} (\lambda=0.5)$	$Q_{\text{aug}} (\lambda=2)$
A (high primary, low constraint)	10	−5	7.5	0
B (moderate primary, high constraint)	9	−1	8.5	7

At $\lambda = 0$ the agent selects A (best primary reward); at $\lambda = 0.5$ the agent selects B, because the low constraint reward of A outweighs its small primary advantage. Neither Q-function alone determines avoidance behavior; it emerges from the $\arg \max$ over their combination. Note that Q_{cost} evaluates the primary-greedy policy from the next step onward, while the agent selects actions according to

Q_{aug} at every step. The constraint reward predictions therefore underestimate the constraint reward the agent actually achieves (since the agent selects via Q_{aug} , which avoids collisions more than the primary-greedy policy that Q_{cost} evaluates). This is conservative for constraint satisfaction: the Lagrangian assigns less safety credit than warranted, biasing the policy toward greater caution. The key advantage is that λ can be varied at *evaluation time* without retraining: sweeping λ over a fixed model traces the objective–constraint Pareto front (Section 3.3).

All regions with the same structure (i.e., the same number of agents) share a single set of Q-function parameters, trained on pooled transitions from all such regions. For the pairwise coordination graph, every region contains exactly two agents, so a single parameter set serves all $\binom{N}{2}$ pairs. This gives $O(1)$ learned models regardless of N , which is critical for scaling to large teams.

Remark (Advantage over Single-Reward Lagrangian). The standard approach of learning a single Q-function for $\tilde{r}_\lambda = r_0 + \lambda r_1$ requires retraining for every λ . The two Q-function formulation instead learns Q_{prim} and Q_{cost} once, and composes them at action selection time via (5). This converts the constrained problem into a family of unconstrained MDPs parameterized by λ , all approximately solvable from a single set of trained parameters, up to the conservative approximation discussed above.

λ -Augmented Variant. An alternative that eliminates this mismatch is to condition the Q-function directly on λ via state augmentation (Calvo-Fullana et al., 2024). Instead of maintaining two Q-functions, we learn a single Q-function $Q_R(s^R, a^R; \lambda)$ that takes the multiplier as an additional input. During training, λ is sampled uniformly from $[0, \lambda_{\text{max}}]$ at the start of each episode, and the Q-function is trained on the augmented reward $r_\lambda = r_0 + \lambda r_1$ via standard Q-learning:

$$y^R = r_0^R + \lambda r_1^R + \gamma \max_{a'^R} Q_R(s'^R, a'^R; \lambda). \quad (6)$$

For each sampled λ , the Q-function learns the optimal policy for that specific tradeoff, so the cost predictions are self-consistent: the policy evaluated by $Q_R(\cdot; \lambda)$ is the same policy that is greedy with respect to $Q_R(\cdot; \lambda)$. At evaluation time, sweeping λ as input traces the Pareto front in the same way as the two-head variant. The tradeoff is that the Q-function must now generalize across both state-action pairs and λ values, which increases the sample complexity of training. This variant feeds its regional Q-tables into the same Max-Sum coordination and parameter sharing pipeline described in Section 3.2, and constitutes a promising direction that we leave for future empirical investigation.

The method requires only that each region R receives a local observation o^R and that the rewards r_0^R, r_1^R satisfy the additive structure (1). The specific instantiation for our experimental domain is described in Section 5.

3.2 Max-Sum Action Coordination

Given the regional Q-tables $\{Q_{\text{aug},R}\}$, agents coordinate actions by solving:

$$\max_{a \in \mathcal{A}} \sum_{R \in \mathcal{C}_R} Q_{\text{aug},R}(s^R, a^R). \quad (7)$$

Direct enumeration is intractable ($|\mathcal{A}|^N$ joint actions), so we exploit the factorization via *Max-Sum message passing* (Farinelli et al., 2008; Rogers et al., 2011) on the factor graph $\mathcal{G}_F$ induced by the coordination graph.

The factor graph is bipartite: *variable nodes* $\{a^i\}_{i \in \mathcal{V}}$ represent agent actions, *factor nodes* $\{Q_{\text{aug},R}\}_{R \in \mathcal{C}_R}$ represent regional Q-functions, and edges connect a^i to $Q_{\text{aug},R}$ when $i \in R$. Max-Sum operates by iteratively exchanging messages between variable and factor nodes.

Factor-to-Variable Messages. For factor Q_R and variable a^i with $i \in R$:

$$M_{R \rightarrow i}^{(n)}(a^i) = \max_{a^{R \setminus i}} \left[Q_{\text{aug},R}(s^R, a^R) + \sum_{k \in R \setminus \{i\}} M_{k \rightarrow R}^{(n-1)}(a^k) \right]. \quad (8)$$

Intuitively, factor Q_R sends agent i the best payoff achievable from region R , maximized over the actions of all other agents in R and accounting for their incoming messages.

Variable-to-Factor Messages. For variable a^i and factor Q_R with $i \in R$:

$$M_{i \rightarrow R}^{(n)}(a^i) = \sum_{\substack{R' \in \mathcal{C}_R: \\ i \in R', R' \neq R}} M_{R' \rightarrow i}^{(n)}(a^i). \quad (9)$$

That is, agent i tells factor Q_R the total value it receives from all *other* regions, summarizing its external commitments. This exclusion prevents double-counting: factor R should not receive evidence about a^i that originated from itself.

Since the pairwise factor graph contains cycles for $N \geq 3$, we apply message damping to stabilize convergence (Murphy et al., 1999); the damping schedule is specified in Appendix 8.2.

After K iterations, each agent selects its action using the accumulated messages [cf. (8)]:

$$a^i = \arg \max_{a^i \in \mathcal{A}^i} \sum_{R: i \in R} M_{R \rightarrow i}^{(K)}(a^i). \quad (10)$$

For pairwise regions ($|R| = 2$), each message requires $O(|\mathcal{A}|)$ operations, giving total complexity $O(\binom{N}{2} |\mathcal{A}| K)$ per step (polynomial in N), compared to $O(|\mathcal{A}|^N)$ for exhaustive search.

3.3 Lagrangian Constraint Handling

We adopt a Lagrangian relaxation to handle the constraints in (2). The Lagrangian is:

$$\mathcal{L}(\pi, \lambda) = V_0^\pi(s_0) + \sum_{j=1}^J \lambda_j \left(V_j^\pi(s_0) - \frac{c_j}{1 - \gamma} \right), \quad (11)$$

where $\lambda_j \geq 0$ is the multiplier for constraint j . Under Assumption (A3), strong duality holds (Paternain et al., 2019): the constrained problem (2) is equivalent to $\min_{\lambda \geq 0} \max_{\pi} \mathcal{L}(\pi, \lambda)$.

For fixed λ , maximizing the Lagrangian reduces to an *unconstrained* MDP with the augmented Q-function (5). the two-head formulation solves this directly: the primary Q-function learns V_0 and the Q-function learns V_j , and λ controls the combination [cf. (5)].

During training, each agent i maintains a local multiplier λ^i , updated via projected dual ascent after each episode based on observed constraint rewards:

$$\lambda^i \leftarrow \Pi_{[0, \lambda_{\max}]} \left(\lambda^i - \eta (\hat{V}_1^i - c_{\text{thresh}}) \right), \quad (12)$$

where $\hat{V}_1^i$ is agent i 's observed per-episode constraint reward, c_{thresh} is the constraint threshold, and η is the dual learning rate (decayed on a slower timescale than the Q-learning rate). Hence, if agent i observes constraint reward below the threshold, its local λ^i increases, shifting the policy toward feasibility. The average multiplier $\bar{\lambda} = \frac{1}{N} \sum_i \lambda^i$ is used in (5) during action selection.

At evaluation time, λ is set externally and swept across a range. For each value, the trained model is evaluated over multiple episodes, recording mean primary objective and constraint cost. This traces the objective–constraint Pareto front from a *single* trained model with no retraining required.

Algorithm 1 summarizes the training procedure.

4 Theoretical Analysis

We establish convergence and approximation properties of CG-CMARL. The analysis addresses two questions: does the algorithm converge to a meaningful solution, and how far is this solution from the true optimum? All proofs are deferred to Appendix 9.

Algorithm 1: CG-CMARL Training – Two-Head Variant (One Episode)

Input: Q-functions $Q_{\text{prim},R}, Q_{\text{cost},R}$ for all $R \in \mathcal{C}_R$, multipliers $\{\lambda^i\}$, replay buffer $\mathcal{D}$

for each time step t do

- Build region observations $\{o^R\}$ from local state projections;
- Compute $Q_{\text{aug},R} \leftarrow Q_{\text{prim},R} + \bar{\lambda} Q_{\text{cost},R}$ for all R ;
- Run damped Max-Sum for K iterations on $\{Q_{\text{aug},R}\}$;
- Select actions via (10);
- Execute a_t , observe s_{t+1} , compute regional rewards $\{r_0^R\}$ and constraint rewards $\{r_1^R\}$;
- Store per-region transitions $(o^R, a^R, r_0^R, r_1^R, o'^R)$ in $\mathcal{D}$;
- Sample batch from $\mathcal{D}$; update $Q_{\text{prim},R}, Q_{\text{cost},R}$ via Eqs. (3)–(4);

Update multipliers $\{\lambda^i\}$ via (12) based on episode constraint rewards;

4.1 Convergence Analysis

We analyze convergence in two stages: first for fixed λ , then for the full primal-dual algorithm. For fixed λ , we require two additional regularity conditions:

(A4) Exploration. Every regional state-action pair (s^R, a^R) is visited infinitely often almost surely.

(A5) Step-Size Conditions. The Q-learning rates satisfy the Robbins-Monro conditions: $\sum_t \alpha_t = \infty, \sum_t \alpha_t^2 < \infty$.

Theorem 4.1 (Regional Q-Learning Convergence). *Under Assumptions (A1)–(A2) and (A4)–(A5), the two-head Q-learning converges almost surely for each region $R \in \mathcal{C}_R$:*

- (i) *The primary Q-function converges to the optimal regional Q-function: $Q_{\text{prim},R,t} \xrightarrow{\text{a.s.}} Q_{0,R}^*$.*
- (ii) *The cost Q-function converges to the constraint-value under the primary greedy policy: $Q_{\text{cost},R,t} \xrightarrow{\text{a.s.}} Q_{\text{cost},R}^\pi$.*

Interpretation. The primary Q-function is updated via standard Q-learning (optimization) and converges to the optimal regional action-values. The constraint Q-function performs TD policy evaluation under the primary Q-function’s greedy policy, which is a contraction with the same modulus γ . Both Q-functions converge independently because Assumption (A1) ensures each regional Bellman operator depends only on local quantities (Tsitsiklis, 1994). Crucially, neither Q-function’s TD target depends on λ , so convergence holds for any trajectory of the multiplier.

We now turn to the full primal-dual algorithm, which updates both Q-functions and multipliers simultaneously. We require an additional condition:

(A6) Timescale Separation. The dual learning rate η satisfies $\eta_t/\alpha_t \rightarrow 0$, so that multiplier updates are slower than Q-learning. Since the multiplier changes once per episode of H steps, the effective ratio $\eta_k/(H\alpha_{kH}) \rightarrow 0$ as training progresses. A standard choice is $\alpha_t = \alpha_0(t+1)^{-3/5}$ and $\eta_t = \eta_0(t+1)^{-9/10}$.

Theorem 4.2 (Primal-Dual Convergence). *Under Assumptions (A1)–(A6), the algorithm achieves:*

- (i) *Q-convergence: Both regional Q-functions converge to their respective optima (primary to $Q_{0,R}^*$, cost to $Q_{\text{cost},R}^\pi$), independently of the multiplier trajectory.*
- (ii) *Dual optimality: The average multiplier $\bar{\lambda}_k = \frac{1}{N} \sum_i \lambda_k^i$ converges to the optimal dual solution: $\bar{\lambda}_k \rightarrow \lambda^* \in \arg \min_{\lambda \geq 0} d(\lambda)$.*

Interpretation. Property (i) is a direct consequence of the two-Q-function architecture: since neither TD target depends on λ , Q-convergence holds without requiring timescale separation. Property (ii) follows because the per-episode dual update (12) performs projected subgradient descent on the dual function. By strong duality (11), convergence to λ^* implies that the induced policy is optimal for (2). The proof leverages two-timescale stochastic approximation theory (Borkar, 1997): the multiplier

changes once per episode while Q-functions update at every step, ensuring the policy stabilizes before the tradeoff shifts.

Corollary 4.3 (Function Approximation). *With Q-functions parameterized by neural networks, the algorithm converges to a neighborhood of a KKT stationary point, with neighborhood size depending on the approximation error ϵ_{NN} .*

This follows from the analysis of nonlinear two-timescale stochastic approximation (Borkar, 1997); the neighborhood radius scales as $O(\epsilon_{\text{NN}}/(1 - \gamma))$ (Appendix 9).

4.2 Approximation Error Decomposition

We characterize the gap between the learned solution and the true optimum. The total error decomposes into four independent sources, each traceable to a specific algorithmic component.

Theorem 4.4 (Compositional Error Bound). *Under Assumptions (A1)–(A3), let Q^* denote the optimal joint Q-function and $\hat{Q} = \sum_R \hat{Q}_R$ the learned factored Q-function. In the tabular case:*

$$\|\hat{Q} - Q^*\|_\infty \leq \underbrace{\frac{\beta}{1 - \gamma}}_{\text{structural}} + \underbrace{\frac{\epsilon_{\text{MS}}}{1 - \gamma}}_{\text{coordination}} + \underbrace{O\left(\frac{1}{\sqrt{n}}\right)}_{\text{sampling}}, \quad (13)$$

where β is the structural error from the factorization approximation (controlled by the coordination graph design), ϵ_{MS} is the Max-Sum coordination error (zero for tree-structured graphs), and $O(1/\sqrt{n})$ is the sampling error from finite data. With function approximation, an additive term ϵ_{NN} for the representation error appears (Corollary 4.3).

Interpretation. The bound reveals that the error sources contribute *additively*: each can be analyzed and controlled independently. In particular, β is controlled by graph design, ϵ_{MS} by graph topology, $O(1/\sqrt{n})$ by sample count, and ϵ_{NN} by network capacity. The $1/(1 - \gamma)$ factors reflect error propagation through the Bellman backup over the infinite horizon. By the simulation lemma (Kearns & Singh, 2002), the policy suboptimality gap is $V^*(s) - V^{\hat{\pi}}(s) \leq 2\epsilon/(1 - \gamma)$ where ϵ is the right-hand side of (13).

A natural question is when the factorization introduces no error:

Proposition 4.5 (Structural Error Characterization). *Under Assumptions (A1)–(A2):*

- (i) *If the regions in $\mathcal{C}_R$ are non-overlapping (each agent belongs to at most one region), then $Q^*(s, a) = \sum_R Q_R^*(s^R, a^R)$ exactly, and $\beta = 0$.*
- (ii) *For overlapping coordination graphs (e.g., pairwise graphs with $N \geq 3$), the structural error satisfies $\beta \geq 0$, with β controlled by the strength of coupling between overlapping regions.*

Interpretation. The structural error arises because independent regional Q-learning optimizes actions within each region separately, ignoring that the same agent’s action appears in multiple overlapping regions. Max-Sum message passing partially compensates for this at action selection time by approximately solving the joint optimization. For the pairwise coordination graph on Simple Spread with $N \geq 3$, $\beta > 0$ in principle, but is small in practice because pairwise collision penalties are weak relative to individual landmark rewards. The factored approach reduces action-selection complexity from $O(|\mathcal{A}|^N)$ to $O(|\mathcal{A}|^2 K)$, and the compositional bound (13) quantifies the cost of this reduction through β and ϵ_{MS} .

5 Experiments

We evaluate CG-CMARL on the Simple Spread environment from the MPE suite (Lowe et al., 2017), where N agents must cover N landmarks in a continuous 2D world $[-1, 1]^2$ while avoiding collisions. This environment satisfies Assumptions (A1)–(A2) (independent transitions, pairwise reward structure). Although the pairwise coordination graph introduces overlapping regions for

$N \geq 3$, so that $\beta > 0$ in principle (Proposition 4.5), the structural error is small in practice because pairwise collision penalties are weak relative to individual landmark rewards.

5.1 Setup

Each pairwise region $R = \{i, k\}$ constructs its observation o^R from locally available information only. Agents in a region do *not* observe out-of-region agents' positions. This ensures truly decentralized execution: each region's Q-network can be evaluated using only locally available data. Note that execution is fully decentralized: action selection requires only local Q-evaluations and Max-Sum messages, with no access to the global state. Training follows a centralized-training, decentralized-execution (CTDE) paradigm: the per-agent counterfactual rewards (14) require all agents' positions to compute the global utility $U(\mathcal{V})$, while the collision cost c^R is purely pairwise. This is standard practice in cooperative MARL and is shared by all our baselines (QMIX, MAPPO, MAPPO-Lagrangian). The full observation structure is given in Appendix 8.2.

To assign credit to individual regions, we use *per-agent counterfactual rewards* (Wolpert & Tumer, 2001). Let $U(\mathcal{V})$ denote the global coverage utility (negative sum of minimum agent-landmark distances) evaluated with agent set $\mathcal{V}$. The primary reward for region R sums the marginal contributions of each agent in R :

$$r_0^R = \sum_{i \in R} [U(\mathcal{V}) - U(\mathcal{V} \setminus \{i\})], \quad (14)$$

where each term measures the coverage improvement attributable to a single agent. Compared to the global counterfactual $U(\mathcal{V}) - U(\mathcal{V} \setminus R)$, which removes the entire region at once, the per-agent variant provides a finer credit assignment signal: it distinguishes the contribution of each agent within the region, avoiding the dilution that occurs when a pair contributes unevenly. The primary reward contains no collision penalty; collision avoidance is handled entirely by the cost head and the Lagrangian multiplier.

We consider $N \in \{3, 4, 6, 10\}$ agents with randomized initial positions for all team sizes. For $N \leq 4$, agents have radius 0.08; for $N > 4$, we auto-scale agent radii to $r = 0.08\sqrt{4/N}$ to prevent overcrowding in the fixed $[-1, 1]^2$ arena. Scaling configurations are summarized in Table 11 (Appendix 10).

All CG-CMARL models implement the two Q-functions as a shared neural network with a common trunk (two FC layers of 128 units) and two output heads, trained with Double Q-learning (Mnih et al., 2015), Polyak target updates ($\tau = 0.005$ every 200 steps), and a replay buffer of size 10^5 . Coordinated exploration is achieved by adding Gaussian noise to the Q-tables before Max-Sum message passing, so that all agents reason about the same perturbed utilities. The constraint threshold is $c_{\text{thresh}} = 0$ (any collision is a violation) and collision penalties are excluded from the primary reward, so collisions are handled entirely via the cost head. For $N \geq 6$, parallel episode collection (4–8 workers) keeps wall-clock training time manageable. Full hyperparameters are in Appendix 10. We evaluate the two-head formulation only; the λ -augmented variant (Section 3.1) is theoretically preferable because the Q-function learns the optimal policy for each λ directly, producing self-consistent constraint predictions without the conservative mismatch of the two-head design. However, the Q-function must generalize across both state-action pairs and λ values, which increases sample complexity. Since the two-head formulation already dominates all baselines under our training budget, we defer empirical evaluation of the augmented variant to future work.

We compare against five methods spanning different design axes: IQL (Independent Q-Learning), which trains per-agent Q-functions without coordination; QMIX (Rashid et al., 2018), which combines agent Q-networks via a monotonic mixing network conditioned on global state; DCG (Böhmer et al., 2020), which uses coordination graphs but without constraint handling; MAPPO, a multi-agent PPO baseline with shared rewards; and MAPPO-Lagrangian, which extends MAPPO with a Lagrangian multiplier for constraint satisfaction. For the value-based methods (IQL, QMIX, DCG), the coverage-safety tradeoff is controlled by the collision-penalty mixing ratio $\alpha \in \{0.0, 0.3, 0.6\}$, so each trained model yields a single operating point. MAPPO is trained with the same mixing ratio configurations

and likewise produces one point per model. MAPPO-Lagrangian learns its own multiplier during training, but this multiplier is frozen in the trained actor and cannot be varied post-hoc, so it too yields a single point. In contrast, CG-CMARL traces the full Pareto front from one trained model by sweeping λ at evaluation time, decoupling the constraint tradeoff from training.

5.2 Pareto Front Analysis

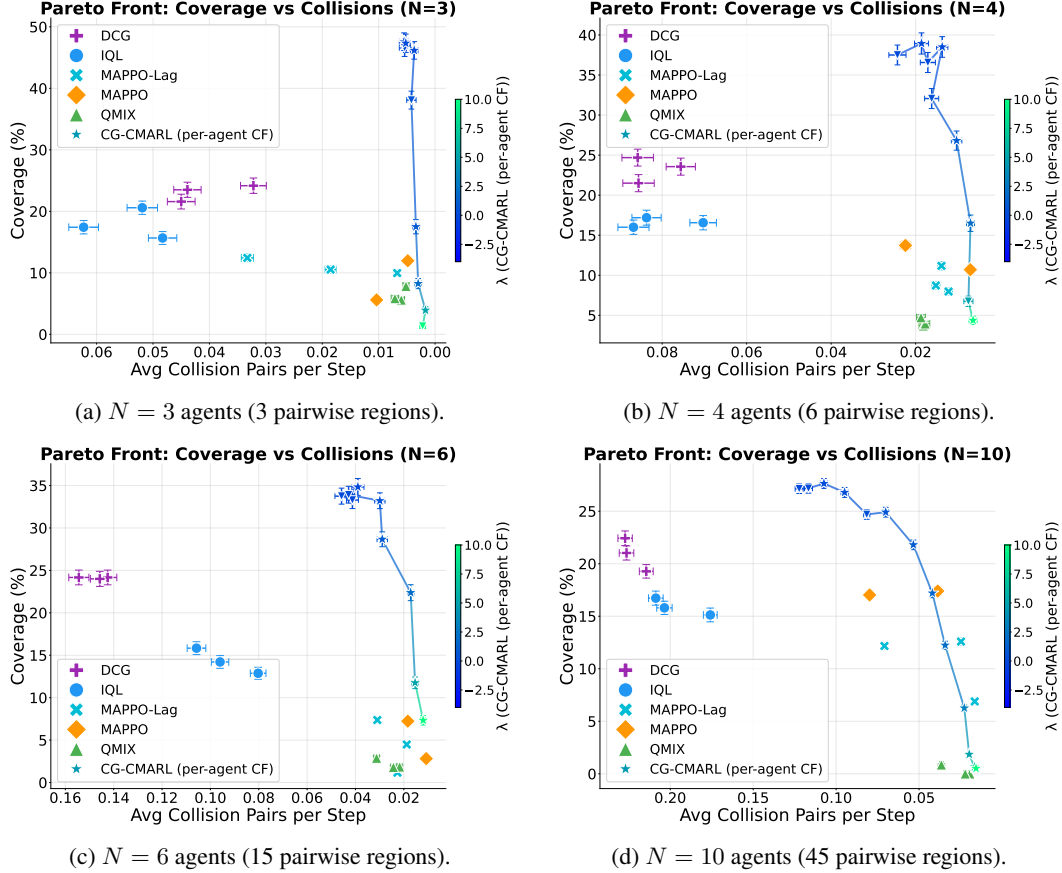


Figure 1: Coverage–safety Pareto fronts on Simple Spread for increasing team sizes. The y -axis, Coverage (%), is the fraction of landmarks with at least one agent within a threshold distance at episode end, averaged over episodes. The x -axis is the average number of overlapping agent pairs per step – i.e., per timestep we count unordered pairs (i, k) with $\|\text{pos}_i - \text{pos}_k\| < 2r_{\text{agent}}$, then average over steps and episodes (lower is better; axis inverted so the upper-right is the ideal corner). CG-CMARL (red) traces the full tradeoff by sweeping λ over a single trained model; stars mark Pareto-optimal operating points and squares mark dominated ones. All other baselines appear as isolated points: IQL, QMIX, DCG, and MAPPO each bake the tradeoff into training via a fixed mixing ratio, while MAPPO-Lagrangian freezes its learned multiplier in the actor. Exact λ values, mixing ratio settings, and numerical results for all operating points are reported in Tables 15–16 (Appendix 10).

Figures 1a–1d show the coverage–safety Pareto fronts for $N \in \{3, 4, 6, 10\}$. CG-CMARL dominates all baselines across team sizes: for any given safety level, the Pareto-optimal points on the λ -sweep (marked with stars in the figures) achieve equal or better coverage than every baseline. Not all points on the sweep are Pareto-optimal; those at extreme λ values (marked with squares) are dominated by moderate- λ points, as discussed below. Even these non-optimal operating points generally dominate or match the baselines.

In the moderate- λ range ($\lambda \in [0.05, 1.0]$), the front traces a clean tradeoff, for example, 70–82% coverage at 93–99% safety for $N = 3$, that occupies a region of the objective space where no baseline operates. This advantage arises from two sources. First, the two Q-function architecture decouples coverage and safety learning, avoiding the interference that occurs when collision penalties are mixed into the reward. Second, Max-Sum coordination enables agents to jointly avoid collisions rather than treating them as independent per-agent penalties.

At very low λ ($\lambda \approx 0$), both coverage and safety degrade simultaneously. Without sufficient collision penalty, multiple agents greedily pursue the same landmark, reducing coverage through redundant assignments while increasing collisions due to spatial proximity. At very high λ ($\lambda \geq 5$), the constraint term dominates Q_{aug} , effectively suppressing the primary signal. Agents then move erratically to avoid any predicted collision, which can paradoxically create new collisions through uncoordinated evasive movements. These extreme operating points are therefore dominated by moderate- λ configurations and do not lie on the Pareto front.

As N increases from 3 to 10, CG-CMARL maintains Pareto dominance. The pairwise coordination graph becomes increasingly dense ($\binom{N}{2}$ regions), amplifying Max-Sum coordination error ϵ_{MS} on the loopy factor graph, and auto-scaled agent radii ($r = 0.08\sqrt{4/N}$ for $N > 4$) reduce the geometric collision cross-section, making the safety constraint easier to satisfy for all methods. At $N = 10$, MAPPO achieves coverage comparable to CG-CMARL at some operating points, though with substantially higher variance across evaluation runs. CG-CMARL’s advantage at this scale lies in providing a reliable, low-variance Pareto front from a single model, rather than requiring multiple independently trained models with uncertain outcomes.

6 Conclusion

Coordination graphs and Lagrangian constraint handling address complementary challenges in cooperative MARL: scalability and safety, respectively. CG-CMARL integrates both by maintaining two Q-functions per region, shared across all pairwise regions via parameter sharing, that separately predict primary reward and constraint reward. Sweeping the Lagrangian multiplier λ at evaluation time traces the full coverage–safety Pareto front from a single trained model. Max-Sum message passing coordinates actions across the factor graph, with complexity polynomial in the number of agents.

The cost Q-function evaluates the primary-greedy policy, yielding a conservative approximation to the true Lagrangian. We additionally discussed a λ -augmented formulation (Section 3.1) that produces self-consistent cost predictions by conditioning the Q-function on λ during training, at the cost of increased sample complexity. Experiments on the Simple Spread cooperative navigation task with up to 10 agents show that CG-CMARL’s Pareto front dominates all baselines, with the advantage most pronounced at moderate team sizes where the coordination graph is sparse relative to the problem structure. We additionally provided convergence guarantees under independent transitions and reward factorization, together with a compositional error bound that decomposes into four interpretable sources.

Limitations and Future Work. The pairwise coordination graph becomes dense for large N ($\binom{N}{2}$ regions), increasing Max-Sum computation. Sparse or dynamic graph topologies (Wang et al., 2022) could reduce this cost. The two-head formulation’s conservative approximation, while empirically effective, means the theoretical convergence guarantees do not strictly apply to the executed policy; the λ -augmented formulation described in Section 3.1 resolves this theoretically at the expense of sample complexity, and its empirical validation is left to future work. A fully decentralized gossip-based multiplier update would extend the framework to settings without a central coordinator, with theoretical support from distributed consensus theory (Nedic & Ozdaglar, 2009). Finally, extending beyond Simple Spread to environments with dependent transitions ($\beta > 0$) would test the practical significance of the structural error bound.

Acknowledgements

This work was supported in part by grant PID2023-146524NB-I00, funded by MCIN/AEI/10.13039/501100011033 and by ERDF/EU. Santiago Amaya-Corredor is supported in part by a predoctoral fellowship (PIPF-FPI) associated with this project, awarded under the 2023 call of the Plan Estatal de Investigación Científica, Técnica y de Innovación 2021–2023.

References

- Leopoldo Agorio, Sean Van Alen, Miguel Calvo-Fullana, Santiago Paternain, and Juan Andrés Bazerque. Multi-agent assignment via state augmented reinforcement learning. In *Proceedings of the 6th Annual Learning for Dynamics & Control Conference*, volume 242 of *Proceedings of Machine Learning Research*, pp. 1202–1213. PMLR, 2024.
- Leopoldo Agorio, Sean Van Alen, Santiago Paternain, Miguel Calvo-Fullana, and Juan Andres Bazerque. Cooperative multi-agent assignment over stochastic graphs via constrained reinforcement learning. *arXiv preprint arXiv:2502.20462*, 2025.
- Stefano V. Albrecht, Filippos Christianos, and Lukas Schäfer. *Multi-Agent Reinforcement Learning: Foundations and Modern Approaches*. MIT Press, 2024.
- Eitan Altman. *Constrained Markov Decision Processes*. Chapman & Hall/CRC, 1999.
- Dimitri Bertsekas. *Convex optimization algorithms*. Athena Scientific, 2015.
- Dimitri Bertsekas. *Reinforcement Learning and Optimal Control*. Athena Scientific, 2019.
- Wendelin Böhmer, Vitaly Kurin, and Shimon Whiteson. Deep coordination graphs. In *International Conference on Machine Learning*, pp. 980–991. PMLR, 2020.
- Vivek S. Borkar. Stochastic approximation with two time scales. *Systems & Control Letters*, 29(5): 291–294, 1997. DOI: 10.1016/S0167-6911(97)90015-3.
- Vivek S. Borkar. An actor-critic algorithm for constrained markov decision processes. *Systems & Control Letters*, 54(3):207–213, 2005.
- Vivek S Borkar and Vivek S Borkar. *Stochastic approximation: a dynamical systems viewpoint*, volume 100. Springer, 2008.
- Craig Boutilier, Richard Dearden, and Moisés Goldszmidt. Stochastic dynamic programming with factored representations. *Artificial Intelligence*, 121(1-2):49–107, 2000.
- Miguel Calvo-Fullana, Santiago Paternain, Luiz F. O. Chamon, and Alejandro Ribeiro. State augmented constrained reinforcement learning: Overcoming the limitations of learning with rewards. *IEEE Transactions on Automatic Control*, 69(7):4275–4290, 2024.
- Wei Duan, Jie Lu, and Junyu Xuan. Group-aware coordination graph for multi-agent reinforcement learning. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence (IJCAI-24)*, pp. 3926–3934. ijcai.org, 2024.
- Eyal Even-Dar and Yishay Mansour. Learning rates for q-learning. *Journal of machine learning Research*, 5(Dec):1–25, 2003.
- Alessandro Farinelli, Alex Rogers, Adrian Petcu, and Nicholas R Jennings. Decentralised coordination of low-power embedded devices using the max-sum algorithm. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pp. 639–646, 2008.
- Shangding Gu, Jakub Grudzien Kuba, Yuanpei Chen, Yali Du, Long Yang, Alois Knoll, and Yaodong Yang. Safe multi-agent reinforcement learning for multi-robot control. *Artificial Intelligence*, 319: 103905, 2023.
- Cong Guan, Tao Jiang, Yi-Chen Li, Zongzhang Zhang, Lei Yuan, and Yang Yu. CUTMAP: Constraining an unconstrained multi-agent policy with offline data. *Neural Networks*, 186:107253, 2025.
- Carlos Guestrin, Daphne Koller, Ronald Parr, and Shobha Venkataraman. Context-specific multiagent coordination and planning with factored MDPs. In *AAAI/IAAI*, pp. 253–259, 2002.

- Nikunj Gupta, James Zachary Hare, Jesse Milzman, Rajgopal Kannan, and Viktor Prasanna. Deep meta coordination graphs for multi-agent reinforcement learning. *arXiv preprint arXiv:2502.04028*, 2025.
- Michael Kearns and Satinder Singh. Near-optimal reinforcement learning in polynomial time. *Machine Learning*, 49(2):209–232, 2002.
- Jelle R. Kok and Nikos Vlassis. Collaborative multiagent reinforcement learning by payoff propagation. *Journal of Machine Learning Research*, 7:1789–1828, 2006.
- Frank R Kschischang, Brendan J Frey, and H-A Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, 2001.
- Ankita Kushwaha, Kiran Ravish, Preeti Lamba, and Pawan Kumar. A survey of safe reinforcement learning and constrained MDPs. *arXiv preprint arXiv:2505.17342*, 2025.
- Ruohong Liu, Yuxin Pan, Linjie Xu, Lei Song, Pengcheng You, Yize Chen, and Jiang Bian. C-morl: Multi-objective reinforcement learning through efficient discovery of pareto front. *arXiv preprint arXiv:2410.02236*, 2024.
- Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in Neural Information Processing Systems*, volume 30, pp. 6380–6391, 2017.
- Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharmashan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015. DOI: 10.1038/nature14236.
- Kevin P Murphy, Yair Weiss, and Michael I Jordan. Loopy belief propagation for approximate inference: An empirical study. In *UAI*, pp. 467–475, 1999.
- Angelia Nedic and Asuman Ozdaglar. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1):48–61, 2009.
- Santiago Paternain, Luiz Chamon, Miguel Calvo-Fullana, and Alejandro Ribeiro. Constrained reinforcement learning has zero duality gap. In *Advances in Neural Information Processing Systems*, volume 32, 2019.
- Santiago Paternain, Miguel Calvo-Fullana, Luiz F. O. Chamon, and Alejandro Ribeiro. Safe policies for reinforcement learning via primal-dual methods. *IEEE Transactions on Automatic Control*, 68(3):1321–1336, 2023.
- Tabish Rashid, Mikayel Samvelyan, Christian Schroeder, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 4295–4304. PMLR, 2018. URL <https://proceedings.mlr.press/v80/rashid18a.html>.
- Alex Rogers, Alessandro Farinelli, Ruben Stranders, and Nicholas R Jennings. Bounded approximate decentralised coordination via the max-sum algorithm. *Artificial Intelligence*, 175(2):730–759, 2011.
- Chapman Siu, Jason Traish, and Richard Yi Da Xu. Dynamic coordination graph for cooperative multi-agent reinforcement learning. In *Asian Conference on Machine Learning*, pp. 438–453. PMLR, 2021.

- Kyunghwan Son, Daewoo Kim, Wan Ju Kang, David E. Hostallero, and Yung Yi. Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning (ICML)*, volume 97 of *Proceedings of Machine Learning Research*, pp. 5887–5896. PMLR, 2019. URL <http://proceedings.mlr.press/v97/son19a.html>.
- Adam Stooke, Joshua Achiam, and Pieter Abbeel. Responsive safety in reinforcement learning by pid lagrangian methods. In *ICML*, 2020.
- 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. 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 (AAMAS)*, pp. 2085–2087, Stockholm, Sweden, 2018. International Foundation for Autonomous Agents and Multiagent Systems. URL <https://dl.acm.org/doi/10.5555/3237383.3238080>.
- Jordan Terry, Benjamin Black, Nathaniel Grammel, Mario Jayakumar, Ananth Hari, Ryan Sullivan, Luis S Santos, Clemens Dieffendahl, Caroline Horsch, Rodrigo Perez-Vicente, et al. Pettingzoo: Gym for multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 34:15032–15043, 2021.
- Chen Tessler, Daniel J Mankowitz, and Shie Mannor. Reward constrained policy optimization. In *ICLR*, 2018.
- John N Tsitsiklis. Asynchronous stochastic approximation and q-learning. *Machine Learning*, 16(3): 185–202, 1994.
- Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.
- Tonghan Wang, Liang Zeng, Weijun Dong, Qianlan Yang, Yang Yu, and Chongjie Zhang. Context-aware sparse deep coordination graphs. In *International Conference on Learning Representations*, 2022.
- David H. Wolpert and Kagan Tumer. Optimal payoff functions for members of collectives. *Advances in Complex Systems*, 4(2–3):265–279, 2001.
- Kaiqing Zhang, Zhuoran Yang, and Tamer Başar. Multi-agent reinforcement learning: A selective overview of theories and algorithms. *Handbook of Reinforcement Learning and Control*, pp. 321–384, 2021.
- Lijun Zhang, Lin Li, Wei Wei, Huizhong Song, Yaodong Yang, and Jiye Liang. Scalable constrained policy optimization for safe multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 37, pp. 138698–138730. Curran Associates, Inc., 2024.
- Xiwen Zhang, Jie Chen, Ming-Gang Gan, and Haoxiang Chen. Influence enhanced sparse coordination graphs for multi-agent reinforcement learning. *Neural Networks*, 188:107454, 2025.
- Xutong Zhao and Yaqi Xie. Multi-level advantage credit assignment for cooperative multi-agent reinforcement learning. In *Proceedings of the 28th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 258 of *Proceedings of Machine Learning Research*, pp. 2971–2979. PMLR, 2025.

Supplementary Materials

The following content was not necessarily subject to peer review.

7 Notation and Definitions

This appendix provides a comprehensive reference for the notation, definitions, and assumptions used throughout the paper.

7.1 Notation Summary

Tables 2–7 collect the notation used in this paper, organized by category.

Table 2: Notation: Agents and state-action spaces

Symbol	Description
N	Number of agents
$\mathcal{V} = \{1, 2, \dots, N\}$	Set of agent indices
$i, k \in \mathcal{V}$	Individual agent indices
$\mathcal{S}^i, \mathcal{A}^i$	Local state/action space of agent i
$s = (s^1, \dots, s^N) \in \mathcal{S}$	Joint state
$a = (a^1, \dots, a^N) \in \mathcal{A}$	Joint action

Table 3: Notation: Coordination graph and regions

Symbol	Description
$\mathcal{G}_R = (\mathcal{V}, \mathcal{C}_R)$	Coordination graph (hypergraph)
$\mathcal{C}_R$	Set of regions (hyperedges)
$R \in \mathcal{C}_R$	A region (subset of agents)
$s^R = (s^i)_{i \in R}, a^R = (a^i)_{i \in R}$	Local projections onto region R
o^R	Region-aware observation for region R
$\mathcal{G}_F$	Factor graph (bipartite) for Max-Sum

Table 4: Notation: Rewards, values, and constraints

Symbol	Description
J	Number of constraints
$r_0(s, a), r_0^R(s^R, a^R)$	Primary reward (global / regional)
c^R	Regional collision cost
$R_{\max}$	Uniform bound on regional rewards
c_j, c_{thresh}	Constraint thresholds
$\gamma \in (0, 1)$	Discount factor
$V_j^\pi(s)$	Value function for reward j under π
$Q_{\text{prim}, R}, Q_{\text{cost}, R}$	Primary / cost head Q-functions
$Q_{\text{aug}, R}$	Augmented Q-function (Eq. (5))

Table 5: Notation: Lagrangian and dual variables

Symbol	Description
$\lambda \geq 0$	Lagrange multiplier
λ^i	Local multiplier of agent i
$\bar{\lambda} = \frac{1}{N} \sum_i \lambda^i$	Average multiplier across agents
$\mathcal{L}(\pi, \lambda)$	Lagrangian function
$d(\lambda)$	Dual function
λ^*	Optimal dual solution

Table 6: Notation: Algorithm parameters

Symbol	Description
$\theta, \bar{\theta}$	Q-network / target network parameters
α_t	Learning rate for Q-updates (fast timescale)
η	Learning rate for multiplier updates (per-episode)
τ	Polyak averaging coefficient for target network
K	Number of Max-Sum iterations
d	Message damping coefficient
ϵ_t	Exploration noise scale
$\mathcal{D}, B$	Replay buffer / mini-batch size
$M_{R \rightarrow i}^{(n)}(a^i)$	Factor-to-variable message
$M_{i \rightarrow R}^{(n)}(a^i)$	Variable-to-factor message

Table 7: Notation: Error terms and bounds

Symbol	Description
β	Structural (factorization) error
ϵ_{MS}	Max-Sum coordination error
ϵ_{NN}	Neural network representation error
ξ	Slater slack (strict feasibility margin)

7.2 Formal Definitions

Definition 7.1 (Markov Decision Process). A Markov Decision Process (MDP) is a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \gamma)$ where $\mathcal{S}$ is a finite state space, $\mathcal{A}$ is a finite action space, $P : \mathcal{S} \times \mathcal{A} \rightarrow \Delta(\mathcal{S})$ is the transition kernel, $r : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ is the reward function, and $\gamma \in (0, 1)$ is the discount factor.

Definition 7.2 (Constrained MDP). A CMDP extends an MDP with J constraint functions: $\mathcal{M}_c = (\mathcal{S}, \mathcal{A}, P, r_0, \{r_j, c_j\}_{j=1}^J, \gamma)$ where r_0 is the primary reward and each (r_j, c_j) defines a constraint $V_j^\pi(s_0) \geq c_j / (1 - \gamma)$.

Definition 7.3 (Coordination Graph). A coordination graph is a hypergraph $\mathcal{G}_R = (\mathcal{V}, \mathcal{C}_R)$ where $\mathcal{V} = \{1, \dots, N\}$ is the vertex set (agents) and $\mathcal{C}_R \subseteq 2^\mathcal{V}$ is the hyperedge set (regions). Each region $R \in \mathcal{C}_R$ is a subset of agents whose joint state-action affects a reward component.

Definition 7.4 (Factor Graph). A factor graph is a bipartite graph $\mathcal{G}_F = (\mathcal{V}_{\text{var}}, \mathcal{V}_{\text{fac}}, \mathcal{E}_F)$. For our coordination problem, variable nodes are agent actions $\{a^i\}$, factor nodes are regional Q-functions $\{Q_{\text{aug}, R}\}$, and the function to optimize is $\sum_R Q_{\text{aug}, R}(o^R, a^R)$.

Definition 7.5 (Regional Bellman Operator). For a region $R \in \mathcal{C}_R$, the regional Bellman operator $\mathcal{T}_R$ acts on Q-functions as

$$(\mathcal{T}_R Q_R)(s^R, a^R) := r^R(s^R, a^R) + \gamma \sum_{s'^R} P(s'^R | s^R, a^R) \max_{a'^R} Q_R(s'^R, a'^R),$$

where $P(s'^R | s^R, a^R) = \prod_{i \in R} P_i(s'^i | s^i, a^i)$ under Assumption (A1). In the implementation, the network takes observations o^R constructed from local states (see Appendix 8.2); the theoretical analysis uses s^R to denote the underlying Markov state.

7.3 Complete Assumption Statements

Assumption A1 (Independent Transitions). $P(s' | s, a) = \prod_{i=1}^N P_i(s'^i | s^i, a^i)$. *Consequence:* The regional Bellman operator depends only on local quantities.

Assumption A2 (Reward Factorization). Each reward signal decomposes additively: $r_j(s, a) = \sum_{R \in \mathcal{C}_R} r_j^R(s^R, a^R)$. We additionally assume bounded rewards: $|r_j^R(s^R, a^R)| \leq R_{\max} < \infty$ for all j, R . *Consequence:* $V_j^\pi(s) = \sum_R V_{j,R}^\pi(s^R)$.

Assumption A3 (Slater’s Condition). There exists $\tilde{\pi} \in \Pi$ and $\xi > 0$ such that $V_j^{\tilde{\pi}}(s_0) \geq c_j/(1 - \gamma) + \xi$ for all j .

Assumption A4 (Exploration). Every regional state-action pair is visited infinitely often a.s.

Assumption A5 (Step-Size Conditions). The Q-learning rates satisfy Robbins-Monro conditions: $\sum_t \alpha_t = \infty$, $\sum_t \alpha_t^2 < \infty$. The dual learning rate η (applied per-episode) satisfies timescale separation: the multiplier changes once per episode of H steps, so the effective ratio $\eta/(H\alpha_{kH}) \rightarrow 0$ as training progresses (see Lemma 9.1). This subsumes Assumption (A6) in the main text, which states the timescale separation condition separately for expository clarity.

7.4 Environment Applicability

Table 8 verifies Assumptions (A1)–(A3) for common MARL environments.

Table 8: Verification of structural assumptions for common MARL environments.

Environment	Transition	Reward	(A1)–(A2)	β
Simple Spread	Ind. kinematics	$\sum_i r_{\text{lm}}^i + \sum_{i < k} r_{\text{col}}^{ik}$	✓	$= 0$ ($N=2$); ≥ 0 ($N \geq 3$)
Cooperative Nav.	Ind. positions	Coverage + collision	✓	$= 0$ ($N=2$); ≥ 0 ($N \geq 3$)
Predator-Prey	Ind. movement	Capture rewards (pairs)	✓	$= 0$ ($N=2$); ≥ 0 ($N \geq 3$)
Warehouse Robots	Ind. motion	Task + collision	✓	$= 0$ ($N=2$); ≥ 0 ($N \geq 3$)

When Is $\beta = 0$? By Proposition 4.5, the structural error vanishes when the coordination graph has *non-overlapping* regions (each agent belongs to at most one region), or equivalently for pairwise graphs with $N = 2$. For pairwise graphs with $N \geq 3$, regions overlap (each agent appears in $N - 1$ pairs), and the independent per-region max no longer equals the joint max. However, β is controlled by the strength of coupling between overlapping regions: when pairwise interaction rewards are small relative to individual terms, β is small in practice. Max-Sum message passing further compensates by approximately solving the joint optimization at action selection time.

Environments Where (A1) Does Not Hold. If collisions cause physical rebounds or agents draw from shared resources, transition dependencies arise. Our approach can still be applied, but $\beta > 0$ with contributions from both action overlap *and* transition coupling.

8 Algorithm Details

This appendix provides the complete pseudocode for CG-CMARL (Section 8.1), implementation details for each component (Section 8.2), and practical hyperparameter choices (Section 10).

8.1 Complete Pseudocode

Algorithm 2: CG-CMARL Training

Input: Shared Q-network Q_θ (trunk + two heads), target network $Q_{\bar{\theta}}$;
Initial local multipliers $\lambda_0^i = 0$ for all $i \in \mathcal{V}$;
Replay buffer $\mathcal{D}$, discount factor $\gamma \in (\lambda_{max})$;
Target update rate $\tau \in (\lambda_{max})$, Max-Sum iterations K ;
Damping coefficient $d \in [\lambda_{max}]$, dual learning rate η ;
Constraint threshold c_{thresh}

for each episode do

for each time step t in episode do

 /* Region-Aware Observations */

foreach region $R = \{i, k\} \in \mathcal{C}_R$ do

 Construct $o^R = [\text{vel}_i, \text{pos}_i, \text{vel}_k, \text{pos}_k, \text{pos}_{\ell_1}, \dots, \text{pos}_{\ell_N}, \text{pos}_k - \text{pos}_i]$;

 /* Augmented Q-Tables */

 Compute average multiplier: $\bar{\lambda} \leftarrow \frac{1}{N} \sum_{i=1}^N \lambda^i$;

foreach region $R \in \mathcal{C}_R$ do

$Q_{\text{aug},R}(o^R, \cdot) \leftarrow Q_{\text{prim},R,\theta}(o^R, \cdot) + \bar{\lambda} Q_{\text{cost},R,\theta}(o^R, \cdot)$;

 /* Coordinated Exploration */

 Add Gaussian noise scaled by exploration rate ϵ_t to each $Q_{\text{aug},R}$;

 /* Damped Max-Sum Action Coordination */

 Initialize messages: $M_{R \rightarrow i}^{(0)}(\cdot) \leftarrow 0$;

$M_{i \rightarrow R}^{(0)}(\cdot) \leftarrow 0$ for all R, i ;

for $n = 1, \dots, K$ do

foreach factor $Q_{\text{aug},R}$ and variable a^i with $i \in R$ do

$\hat{M}_{R \rightarrow i}^{(n)}(a^i) \leftarrow \max_{a^{R \setminus i}} [Q_{\text{aug},R}(o^R, a^R) + \sum_{k \in R \setminus \{i\}} M_{k \rightarrow R}^{(n-1)}(a^k)]$;

 Damp: $M_{R \rightarrow i}^{(n)} \leftarrow (1 - d) \hat{M}_{R \rightarrow i}^{(n)} + d M_{R \rightarrow i}^{(n-1)}$;

foreach variable a^i and factor $Q_{\text{aug},R}$ with $i \in R$ do

$M_{i \rightarrow R}^{(n)}(a^i) \leftarrow \sum_{R' \neq R: i \in R'} M_{R' \rightarrow i}^{(n)}(a^i)$;

 Each agent selects: $a_t^i \leftarrow \arg \max_{a^i \in \mathcal{A}^i} \sum_{R: i \in R} M_{R \rightarrow i}^{(K)}(a^i)$;

 /* Environment Interaction */

 Execute joint action a_t , observe s_{t+1} ;

 Compute per-agent counterfactual rewards $\{r_0^R\}$ via (14) and constraint rewards $\{r_1^R\}$;

 /* Experience Storage */

foreach region $R \in \mathcal{C}_R$ do

 Store transition $(o^R, a^R, r_0^R, r_1^R, o'^R)$ in $\mathcal{D}$;

 /* Q-Network Update (Both Heads) */

 Sample mini-batch from $\mathcal{D}$;

foreach transition $(o^R, a^R, r_0^R, r_1^R, o'^R)$ in batch do

 Primary target: $y_{\text{prim}}^R \leftarrow r_0^R + \gamma Q_{\text{prim},R,\bar{\theta}}(o'^R, \arg \max_{a'} Q_{\text{prim},R,\theta}(o'^R, a'))$;

 Cost target: $y_{\text{cost}}^R \leftarrow r_1^R + \gamma Q_{\text{cost},R,\bar{\theta}}(o'^R, \arg \max_{a'} Q_{\text{prim},R,\theta}(o'^R, a'))$;

 Update θ by minimizing $\frac{1}{2} (Q_{\text{prim},R,\theta} - y_{\text{prim}}^R)^2 + \frac{1}{2} (Q_{\text{cost},R,\theta} - y_{\text{cost}}^R)^2$;

 Soft-update target: $\bar{\theta} \leftarrow \tau \theta + (1 - \tau) \bar{\theta}$;

 /* Per-Episode Multiplier Update */

foreach agent $i \in \mathcal{V}$ do

$\lambda^i \leftarrow \Pi_{[0, \lambda_{\text{max}}]} (\lambda^i - \eta (\hat{V}_1^i - c_{\text{thresh}}))$;

Algorithm 2 presents the full CG-CMARL training procedure. At each time step, the algorithm builds region-aware observations, coordinates actions via damped Max-Sum on the augmented Q-tables, collects experience, and updates both heads of the shared Q-network. At the end of each episode, local Lagrange multipliers are updated via dual ascent based on observed constraint rewards.

Evaluation Protocol. At evaluation time, the trained Q-network is frozen and the Lagrangian multiplier λ is set externally (not learned). For each value $\lambda \in \{0, 0.1, 0.2, 0.5, 1, 2, 5, 10\}$, the model is evaluated over multiple episodes, recording average coverage and collision rate. This procedure traces the coverage–safety Pareto front from a single trained model without retraining.

8.2 Component Details

8.2.1 Two-Head Q-Network

The shared Q-network Q_θ consists of a common trunk and two output heads, all parameterized by θ :

- *Shared trunk*: Two fully connected layers (128 units each) with ReLU activations, mapping the region observation o^R to a shared feature representation $h^R \in \mathbb{R}^{128}$.
- *Primary head* $Q_{\text{prim},R} : \mathbb{R}^{|o^R|} \rightarrow \mathbb{R}^{|A^R|}$: A linear layer mapping h^R to one Q-value per joint action in the region, trained with coverage reward r_0^R .
- *Cost head* $Q_{\text{cost},R} : \mathbb{R}^{|o^R|} \rightarrow \mathbb{R}^{|A^R|}$: A linear layer mapping h^R to one Q-value per joint action, trained with collision cost c^R .

The key design choice is that the cost head’s bootstrap target uses the *primary* head’s greedy action [cf. (4)], so that it learns to predict collision costs under the policy induced by coverage optimization. This ensures that the cost predictions remain meaningful when composed with the primary head at evaluation time.

Parameter Sharing. All pairwise regions share a single Q-network. Since every region $R = \{i, k\}$ contains exactly two agents, the observation and action dimensions are identical across regions. Transitions from all $\binom{N}{2}$ regions are pooled into a single replay buffer and used to train the shared network. This gives $O(1)$ learned parameters regardless of team size N .

Observation Construction. For each pairwise region $R = \{i, k\}$, the observation o^R is constructed from locally available information:

$$o^R = \left[\underbrace{(\text{vel}_i, \text{pos}_i, \text{vel}_k, \text{pos}_k)}_{\text{agent states}}; \underbrace{(\text{pos}_{\ell_1}, \dots, \text{pos}_{\ell_N})}_{\text{landmark positions}}; \underbrace{(\text{pos}_k - \text{pos}_i)}_{\text{internal relative}} \right].$$

Agents in a region do *not* observe out-of-region agents’ positions, ensuring truly decentralized execution. The observation dimension for the pairwise case is $4|R| + 2N + 2|R|(|R| - 1)$.

8.2.2 Max-Sum Action Coordination

Max-Sum solves the combinatorial coordination problem $\max_{a \in \mathcal{A}} \sum_R Q_{\text{aug},R}(o^R, a^R)$ via message passing on the factor graph induced by the coordination graph.

Factor Graph Construction. The factor graph $\mathcal{G}_F = (\mathcal{V}_{\text{var}}, \mathcal{V}_{\text{fac}}, \mathcal{E}_F)$ is bipartite:

- *Variable nodes* $\mathcal{V}_{\text{var}} = \{a^i\}_{i \in \mathcal{V}}$: one per agent, representing its action choice.
- *Factor nodes* $\mathcal{V}_{\text{fac}} = \{Q_{\text{aug},R}\}_{R \in \mathcal{C}_R}$: one per region, representing the regional augmented Q-function.
- *Edges* $\mathcal{E}_F$: variable a^i connects to factor $Q_{\text{aug},R}$ if and only if $i \in R$.

Message Semantics. The two message types have complementary roles:

- $M_{R \rightarrow i}^{(n)}(a^i)$: The best payoff achievable in region R if agent i plays action a^i , maximized over the actions of all other agents in R and accounting for their incoming messages from other regions.
- $M_{i \rightarrow R}^{(n)}(a^i)$: The total value agent i receives from all regions *other than* R for action a^i . This exclusion prevents double-counting: factor R should not receive evidence about a^i that originated from itself.

Damping. Since the pairwise factor graph contains cycles for $N \geq 3$ agents, we apply message damping to stabilize convergence. After computing each factor-to-variable message $\hat{M}_{R \rightarrow i}^{(n)}$, the damped message is:

$$M_{R \rightarrow i}^{(n)} = (1 - d) \hat{M}_{R \rightarrow i}^{(n)} + d M_{R \rightarrow i}^{(n-1)}, \quad (15)$$

with damping coefficient $d = 0.3$. This is standard practice for loopy belief propagation (Murphy et al., 1999) and ensures that messages evolve smoothly across iterations.

Coordinated Exploration. During training, Gaussian noise scaled by the current exploration rate ϵ_t is added to the augmented Q-tables *before* message passing begins. That is, all agents reason about the same perturbed utilities during Max-Sum, preserving coordination structure. This contrasts with ϵ -greedy exploration applied *after* action selection, which would break coordination by independently randomizing individual agents' actions.

Complexity. Each factor-to-variable message requires maximizing over the actions of all agents in R except i . For pairwise regions ($|R| = 2$), this is $O(|\mathcal{A}^k|)$ per message. The total complexity per time step is:

$$\text{Total} = O(|\mathcal{C}_R| |R| |\mathcal{A}| K) = O\left(\binom{N}{2} 2 |\mathcal{A}| K\right) = O(N^2 K |\mathcal{A}|),$$

compared to $O(|\mathcal{A}|^N)$ for exhaustive search—an exponential improvement.

Convergence. For tree-structured factor graphs, Max-Sum converges in at most $\text{diam}(\mathcal{G}_F)$ iterations to the exact global optimum (Kschischang et al., 2001). For cyclic graphs (which arise for pairwise coordination with $N \geq 3$), convergence is not guaranteed in general. We run a fixed K iterations with damping; empirically, $K = 10$ with $d = 0.3$ yields near-optimal solutions on pairwise graphs with up to $N = 10$ agents.

8.2.3 Per-Agent Counterfactual Rewards for Credit Assignment

To assign credit to individual regions, we use per-agent counterfactual rewards (Wolpert & Tumer, 2001). Let $U(\mathcal{V})$ denote the global coverage utility (negative sum of minimum agent-landmark distances) evaluated with agent set $\mathcal{V}$. The primary reward for region R sums the marginal contributions of each agent in R :

$$r_0^R = \sum_{i \in R} [U(\mathcal{V}) - U(\mathcal{V} \setminus \{i\})], \quad (16)$$

where each term measures the coverage improvement attributable to a single agent. This per-agent decomposition provides a finer credit assignment signal than removing the entire region at once, and distinguishes the contribution of each agent within the region. With `local_ratio` = 0.0, the primary reward contains no collision penalty; collision avoidance is handled entirely by the cost head and the Lagrangian multiplier.

8.2.4 Lagrangian Multiplier Update

Each agent i maintains a local multiplier $\lambda^i \in [0, \lambda_{\max}]$, updated at the end of each episode based on observed constraint rewards.

Dual Ascent Update. After each episode, agent i updates its multiplier via:

$$\lambda^i \leftarrow \Pi_{[0, \lambda_{\max}]} \left(\lambda^i - \eta (\hat{V}_1^i - c_{\text{thresh}}) \right), \quad (17)$$

where $\hat{V}_1^i$ is agent i 's observed per-episode constraint reward, c_{thresh} is the constraint threshold, and η is the dual learning rate. The projection to $[0, \lambda_{\max}]$ prevents multiplier divergence.

Interpretation. If agent i observes constraint reward below the threshold ($\hat{V}_1^i < c_{\text{thresh}}$), λ^i increases, which in turn increases the weight on the cost head in the augmented Q-function (5), shifting the policy toward safer actions. Conversely, if constraint reward exceeds the threshold, λ^i decreases, allowing the agent to prioritize coverage.

Central Averaging. The multiplier used in (5) during action selection is the team average $\bar{\lambda} = \frac{1}{N} \sum_{i=1}^N \lambda^i$. This averaging serves two purposes: (i) it ensures all regions use a common augmented Q-function, which is necessary for Max-Sum to optimize a consistent objective; and (ii) it pools local constraint observations across agents, providing a more accurate estimate of team-wide constraint satisfaction.

Remark (Toward Fully Decentralized Multiplier Updates). The current implementation computes $\bar{\lambda}$ centrally during training. A fully decentralized variant would replace this central averaging step with gossip consensus (Nedic & Ozdaglar, 2009): each agent exchanges its local λ^i with communication neighbors and computes a weighted average using doubly stochastic weights. The theoretical analysis in Section 4.1 supports this extension, as the primal-dual convergence guarantees hold under gossip-based consensus (Borkar, 1997). $\triangle$

Timescale Separation. The dual learning rate η is set on a slower timescale than the Q-network learning rate α , satisfying $\eta \ll \alpha$. This ensures that, from the perspective of Q-learning, the multipliers appear nearly constant—the Q-functions can track the optimal values for the current $\bar{\lambda}$ before the multiplier changes. In practice, the multiplier is updated once per episode, while the Q-network is updated at every time step.

8.3 Implementation Details

Network Architecture. The shared two-head Q-network Q_θ consists of:

- *Input:* Region observation o^R (dimension depends on N ; see Table 11)
- *Trunk:* 2 fully connected layers $\times$ 128 units, ReLU activations
- *Primary head:* Linear layer $\rightarrow |\mathcal{A}^R| = |\mathcal{A}^i| \times |\mathcal{A}^k|$ outputs
- *Cost head:* Linear layer $\rightarrow |\mathcal{A}^R|$ outputs

Note that $\bar{\lambda}$ is *not* an input to the network. The multiplier enters only at action selection time through the augmented Q-function (5), which is what enables Pareto front evaluation from a single trained model.

Action Discretization. For environments with continuous actions, we discretize each dimension into 5 bins, yielding $|\mathcal{A}^i| = 25$ actions per agent (for 2D action spaces). For pairwise regions, this gives $|\mathcal{A}^R| = 25 \times 25 = 625$ joint actions per region. The discretization enables Max-Sum coordination while maintaining connections to the tabular convergence theory (Theorem 4.1).

Exploration. We use Gaussian noise added to the augmented Q-tables before Max-Sum message passing. The noise scale decays as $\epsilon_t = \max(0.05, 0.9 - 0.85t/T_{\text{decay}})$. This ensures coordinated exploration: all agents reason about the same perturbed Q-values, so their actions remain jointly coherent even during exploration.

Double Q-Learning and Target Networks. Both heads use Double Q-learning (Van Hasselt et al., 2016) to reduce overestimation bias. A target network $Q_{\bar{\theta}}$ is maintained via Polyak averaging:

$\bar{\theta} \leftarrow \tau\theta + (1 - \tau)\bar{\theta}$ with $\tau = 0.005$, applied every 200 time steps. The cost head’s bootstrap target uses the primary head’s greedy action [cf. (4)], ensuring consistency between the two heads.

Replay Buffer. Transitions are stored per-region: each entry is $(o^R, a^R, r_0^R, c^R, o'^R)$. Since all pairwise regions share a single network, transitions from all $\binom{N}{2}$ regions are pooled into one buffer of size $|\mathcal{D}| = 10^5$. Mini-batches of size $B = 64$ are sampled uniformly for training.

Scaling Configurations. Table 11 summarizes the observation dimensions and number of pairwise regions for each team size.

Table 9: Scaling configurations for different team sizes.

N	Agent size	Pairwise regions	Obs. dim	Scenario
3	0.080	3	18	Random
4	0.080	6	20	Random
6	0.065	15	24	Random
10	0.051	45	32	Random

Hyperparameters. Table 10 summarizes the hyperparameter values used in experiments.

Table 10: Hyperparameter values for CG-CMARL experiments.

Parameter	Symbol	Value
Discount factor	γ	0.99
Target network update rate	τ	0.005
Target update frequency	—	every 200 steps
Max-Sum iterations	K	10
Max-Sum damping coefficient	d	0.3
Replay buffer size	$ \mathcal{D} $	10^5
Mini-batch size	B	64
Q-network learning rate	α	10^{-3} (Adam)
Dual learning rate	η	10^{-2}
Constraint threshold	c_{thresh}	0
Multiplier projection range	$[\lambda_{\min}, \lambda_{\max}]$	$[0, \lambda_{\max}]$
Exploration decay period	T_{decay}	10^5 steps
Minimum exploration rate	$\epsilon_{\min}$	0.05
Local ratio / mixing ratio	local_ratio	0.0
Trunk hidden layers	—	2×128 , ReLU
Actions per dimension	—	5 (total $ \mathcal{A}^i = 25$)

Parallel Episode Collection. For $N \geq 6$ agents, wall-clock training time becomes significant due to the $O(N^2)$ pairwise regions. To mitigate this, we use parallel episode collection with 4–8 workers, each running independent environment instances and contributing transitions to the shared replay buffer.

9 Proofs

This appendix contains complete proofs of all theoretical results. Section 9.1 proves Q-learning convergence (Theorem 4.1). Section 9.2 proves primal-dual convergence (Theorem 4.2). Section 9.3 establishes the error decomposition (Theorem 4.4 and Proposition 4.5). Section 9.4 proves the policy suboptimality bound.

9.1 Proof of Theorem 4.1: Regional Q-Learning Convergence

We prove that the two-head regional Q-learning converges for any fixed multiplier $\lambda \geq 0$. The two heads learn different quantities—the primary head performs Q-learning (optimization), while the cost head performs TD policy evaluation—so we treat each separately.

9.1.1 Primary Head Convergence

The primary head learns $Q_{\text{prim},R}$ via standard Q-learning with reward r_0^R .

Proof. The proof proceeds in three steps: (i) the regional Bellman operator is well-defined under Assumption (A1), (ii) it is a contraction, and (iii) stochastic approximation theory guarantees convergence.

Step 1: The regional Bellman operator is well-defined.

For region $R \in \mathcal{C}_R$, define the regional Bellman operator $\mathcal{T}_{0,R} : \mathbb{R}^{|S^R| \times |\mathcal{A}^R|} \rightarrow \mathbb{R}^{|S^R| \times |\mathcal{A}^R|}$ by

$$(\mathcal{T}_{0,R}Q)(s^R, a^R) := r_0^R(s^R, a^R) + \gamma \sum_{s'^R \in S^R} P(s'^R | s^R, a^R) \max_{a'^R \in \mathcal{A}^R} Q(s'^R, a'^R). \quad (18)$$

Under Assumption (A1) (independent transitions), the marginal transition probability satisfies

$$P(s'^R | s, a) = P(s'^R | s^R, a^R) = \prod_{i \in R} P_i(s'^i | s^i, a^i),$$

where the first equality holds because agents outside R do not affect states inside R . Hence, $\mathcal{T}_{0,R}$ depends only on (s^R, a^R) and is well-defined as an operator on regional Q-functions.

Step 2: The operator is a γ -contraction.

We show $\|\mathcal{T}_{0,R}Q_1 - \mathcal{T}_{0,R}Q_2\|_\infty \leq \gamma\|Q_1 - Q_2\|_\infty$ for any Q_1, Q_2 . Using definition (18):

$$\begin{aligned} |(\mathcal{T}_{0,R}Q_1 - \mathcal{T}_{0,R}Q_2)(s^R, a^R)| &= \gamma \left| \sum_{s'^R} P(s'^R | s^R, a^R) \left[\max_{a'^R} Q_1(s'^R, a'^R) - \max_{a'^R} Q_2(s'^R, a'^R) \right] \right| \\ &\leq \gamma \sum_{s'^R} P(s'^R | s^R, a^R) \left| \max_{a'^R} Q_1(s'^R, a'^R) - \max_{a'^R} Q_2(s'^R, a'^R) \right|. \end{aligned} \quad (19)$$

By non-expansiveness of the max operator, $|\max_a f(a) - \max_a g(a)| \leq \max_a |f(a) - g(a)|$, we obtain

$$\left| \max_{a'^R} Q_1(s'^R, a'^R) - \max_{a'^R} Q_2(s'^R, a'^R) \right| \leq \|Q_1 - Q_2\|_\infty.$$

Substituting into (19) and using $\sum_{s'^R} P(s'^R | s^R, a^R) = 1$:

$$|(\mathcal{T}_{0,R}Q_1 - \mathcal{T}_{0,R}Q_2)(s^R, a^R)| \leq \gamma\|Q_1 - Q_2\|_\infty.$$

Taking the supremum over (s^R, a^R) yields the contraction property.

Step 3: Stochastic approximation convergence.

The primary head update (3) can be written as

$$Q_{R,t+1}(s_t^R, a_t^R) = (1 - \alpha_t)Q_{R,t}(s_t^R, a_t^R) + \alpha_t (\mathcal{T}_{0,R}Q_{R,t}(s_t^R, a_t^R) + w_t^R), \quad (20)$$

where $w_t^R := r_0^R + \gamma \max_{a'} Q_{R,t}(s_{t+1}^R, a') - \mathcal{T}_{0,R}Q_{R,t}(s_t^R, a_t^R)$ is the noise term. Let $\mathcal{F}_t = \sigma(s_0, a_0, \dots, s_t, a_t)$ denote the filtration generated by the trajectory. Then $\mathbb{E}[w_t^R | \mathcal{F}_t] = 0$ (martingale difference sequence) and $\mathbb{E}[|w_t^R|^2 | \mathcal{F}_t] \leq C$ for some constant C depending on $R_{\max}$ and γ (bounded variance, using Assumption (A2)).

By Assumption A5, the step sizes satisfy $\sum_t \alpha_t = \infty$ and $\sum_t \alpha_t^2 < \infty$. By Assumption A4, each state-action pair is visited infinitely often. These conditions, combined with the contraction property of $\mathcal{T}_{0,R}$, satisfy the requirements of Tsitsiklis (1994, Theorem 1). Therefore, $Q_{\text{prim},R,t} \rightarrow Q_{0,R}^*$ almost surely, where $Q_{0,R}^*$ is the unique fixed point of $\mathcal{T}_{0,R}$. $\square$

9.1.2 Cost Head Convergence

The cost head learns $Q_{\text{cost},R}$ via TD evaluation: the bootstrap target uses the *primary head's* greedy action (4), not the cost head's own greedy action. Hence, the cost head performs *policy evaluation* under the primary head's policy, not optimization.

Proof. For a fixed primary policy $\pi_{\text{prim},R}(s^R) := \arg \max_{a^R} Q_{\text{prim},R}^*(s^R, a^R)$, define the regional policy evaluation operator:

$$(\mathcal{T}_{\pi,R}Q)(s^R, a^R) := c^R(s^R, a^R) + \gamma \sum_{s'^R} P(s'^R | s^R, a^R) Q(s'^R, \pi_{\text{prim},R}(s'^R)). \quad (21)$$

This operator is a γ -contraction in $\|\cdot\|_\infty$ by the same argument as Step 2 above, with the max replaced by evaluation at a fixed action. Indeed, the bound is tighter: the operator is linear in Q for fixed π , so

$$\begin{aligned} |(\mathcal{T}_{\pi,R}Q_1 - \mathcal{T}_{\pi,R}Q_2)(s^R, a^R)| &= \gamma \left| \sum_{s'^R} P(s'^R | s^R, a^R) [Q_1(s'^R, \pi(s'^R)) - Q_2(s'^R, \pi(s'^R))] \right| \\ &\leq \gamma \|Q_1 - Q_2\|_\infty. \end{aligned} \quad (22)$$

The cost head update (4) uses $\arg \max_{a'} Q_{\text{prim},R,\theta}(o'^R, a')$ as the bootstrap action. By the convergence of the primary head (shown above) and the timescale separation, the primary head's greedy policy stabilizes. Consequently, the cost head's update becomes a stochastic approximation for $\mathcal{T}_{\pi,R}$, and the same argument as Step 3 yields $Q_{\text{cost},R,t} \rightarrow Q_{\text{cost},R}^\pi$ almost surely, where $Q_{\text{cost},R}^\pi$ is the unique fixed point of $\mathcal{T}_{\pi,R}$ —that is, the expected discounted collision cost under the primary policy. $\square$

9.1.3 Independence Across Regions

Although all regions share data from the same trajectory (s_t, a_t, s_{t+1}) , the updates for different regions are decoupled: the update for Q_R depends only on $(s_t^R, a_t^R, s_{t+1}^R)$. Hence, convergence of each regional Q-function (both heads) can be established independently. With parameter sharing, all pairwise regions train a single network; the convergence guarantee then applies to the shared network in the tabular sense, with the pooled transitions providing the exploration coverage required by Assumption A4.

9.2 Proof of Theorem 4.2: Primal-Dual Convergence

We prove the two-part convergence result for the full primal-dual algorithm. In CG-CMARL, each agent i maintains a local multiplier λ^i updated per-episode via dual ascent (12), and the team average $\bar{\lambda} = \frac{1}{N} \sum_i \lambda^i$ is used for action selection. The proof shows that (i) Q-functions track the optimal values for the current $\bar{\lambda}$, and (ii) $\bar{\lambda}$ converges to the optimal dual solution.

9.2.1 Preliminary: Step-Size Conditions

Lemma 9.1 (Step-Size Conditions). *Let $\alpha_t = \alpha_0/(t+1)^{\rho_\alpha}$ and $\eta_k = \eta_0/(k+1)^{\rho_\eta}$ (where k indexes episodes and t indexes time steps) with $0.5 < \rho_\alpha < \rho_\eta \leq 1$. Then:*

- (i) $\sum_t \alpha_t = \sum_k \eta_k = \infty$
- (ii) $\sum_t \alpha_t^2 < \infty$ and $\sum_k \eta_k^2 < \infty$

(iii) The multiplier updates (per-episode) are on a slower timescale than Q -updates (per-step): since each episode contains H steps, the effective ratio is $\eta_k / (H \alpha_{kH}) \rightarrow 0$.

Proof. Properties (i) and (ii) follow from the p -series: $\sum_t (t+1)^{-\rho}$ diverges for $\rho \leq 1$ and $\sum_t (t+1)^{-2\rho}$ converges for $\rho > 0.5$. Property (iii) holds because $\eta_k / \alpha_{kH} = (\eta_0 / \alpha_0) \cdot (kH+1)^{\rho_\alpha} / (k+1)^{\rho_\eta}$, which tends to zero when $\rho_\eta > \rho_\alpha$ (the episode index k grows slower than the step index $t = kH$). $\square$

In practice, CG-CMARL uses a fixed dual learning rate η with per-episode updates and a decaying Q-learning rate, which achieves the required timescale separation: Q-networks update many times per episode while the multiplier updates once.

9.2.2 Part (i): Q-Convergence on the Fast Timescale

Proof. On the fast timescale (per-step Q-updates), the multiplier $\bar{\lambda}$ evolves slowly because it changes only once per episode. By the “quasi-static” analysis of two-timescale stochastic approximation (Borkar & Borkar, 2008, Theorem 2, Chapter 6), the fast iterate (Q-functions) tracks the equilibrium of the fast dynamics for the slowly-varying slow iterate (multiplier).

Formally, define the “frozen” primary Q-function $Q_{\text{prim},R}^*$ as the fixed point of $\mathcal{T}_{0,R}$, and the frozen cost Q-function $Q_{\text{cost},R}^\pi$ as the fixed point of $\mathcal{T}_{\pi,R}$ under the primary greedy policy. By Theorem 4.1, both heads converge when $\bar{\lambda}$ is fixed. The two-timescale theory extends this to show that when $\bar{\lambda}$ varies slowly, the Q-functions remain close to their frozen optima:

$$\limsup_{t \rightarrow \infty} \|Q_{\text{prim},R,t} - Q_{0,R}^*\|_\infty = 0 \quad \text{a.s.}$$

and similarly for the cost head. Since $\bar{\lambda}$ enters only at action selection time through (5) and does not affect the individual head training targets, the primary and cost heads converge independently of $\bar{\lambda}$. $\square$

Remark 9.2 (Decoupling from λ). A key property of the two-head architecture is that $\bar{\lambda}$ does not appear in the TD targets (3) or (4). Both heads are trained with fixed reward signals (r_0^R and c^R respectively). The multiplier $\bar{\lambda}$ enters only at action selection time through the augmented Q-function (5). Consequently, the Q-convergence in Part (i) holds for *any* trajectory of $\bar{\lambda}$, without requiring timescale separation for the Q-learning convergence itself. The timescale separation is needed only for Part (ii), to ensure the policy induced by $\bar{\lambda}$ stabilizes before the multiplier updates. $\triangle$

9.2.3 Part (ii): Dual Optimality

Proof. We show that the average multiplier $\bar{\lambda}_k := \frac{1}{N} \sum_{i=1}^N \lambda_k^i$ (where k indexes episodes) converges to the optimal dual solution λ^* .

Step 1: Characterize the average dynamics.

Summing the per-agent dual update (12) over all agents and dividing by N :

$$\bar{\lambda}_{k+1} = \frac{1}{N} \sum_{i=1}^N \Pi_{\lambda_{\max}} (\lambda_k^i + \eta (c_k^i - c_{\text{thresh}})). \quad (23)$$

The projection to $[0, \lambda_{\max}]$ is a projection onto the compact set $\Lambda := [0, \lambda_{\max}]$. For the theoretical analysis, we note that under Assumption (A3) (Slater’s condition), the optimal dual variables λ^* are bounded. The range $[0, \lambda_{\max}]$ must be chosen large enough to contain λ^* ; we assume this holds.

Since all agents are initialized identically ($\lambda_0^i = 0$) and observe i.i.d. collision costs (by the symmetry of the pairwise coordination graph and parameter sharing), the local multipliers remain close to

each other throughout training. In particular, $\lambda_k^i \approx \bar{\lambda}_k$ for all i after the first few episodes. We can therefore approximate the average dynamics as:

$$\bar{\lambda}_{k+1} \approx \Pi_{\Lambda} \left[\bar{\lambda}_k + \eta \left(\frac{1}{N} \sum_{i=1}^N c_k^i - c_{\text{thresh}} \right) \right], \quad (24)$$

where Π_{Λ} denotes projection onto $\Lambda = [0, 1]$.

Step 2: Identify the mean field.

The term $\frac{1}{N} \sum_{i=1}^N c_k^i$ is the team-average collision cost in episode k . Under the policy $\pi_{\bar{\lambda}_k}$ (induced by the augmented Q-function with multiplier $\bar{\lambda}_k$), the expected team-average collision cost is

$$\mathbb{E}_{\pi_{\bar{\lambda}_k}} \left[\frac{1}{N} \sum_{i=1}^N c_k^i \right] = (1 - \gamma) V_1^{\pi_{\bar{\lambda}_k}}(s_0),$$

where V_1^{π} is the value function for the constraint reward (collision cost) under policy π , and the $(1 - \gamma)$ factor converts the discounted sum to an average-reward scale.

The mean-field ODE associated with (24) is therefore:

$$\dot{\lambda} = \Pi_{T_{\Lambda}(\lambda)} [(1 - \gamma) V_1^{\pi_{\lambda}}(s_0) - c_{\text{thresh}}], \quad (25)$$

where $\Pi_{T_{\Lambda}(\lambda)}$ denotes projection onto the tangent cone of Λ at λ .

Step 3: Connection to dual descent.

The Lagrangian of problem (2) for a single constraint ($J = 1$) is

$$\mathcal{L}(\pi, \lambda) = V_0^{\pi}(s_0) + \lambda \left(V_1^{\pi}(s_0) - \frac{c_1}{1 - \gamma} \right).$$

The dual function is $d(\lambda) := \max_{\pi} \mathcal{L}(\pi, \lambda)$, and a subgradient of d at λ is

$$g(\lambda) = V_1^{\pi_{\lambda}}(s_0) - \frac{c_1}{1 - \gamma}.$$

Comparing with (25), the multiplier update performs projected subgradient descent on $d(\lambda)$ (up to the $(1 - \gamma)$ scaling, which affects the rate but not the fixed point).

Step 4: Convergence.

By Assumption (A3) (Slater's condition), the dual function $d(\lambda)$ is coercive: $d(\lambda) \rightarrow \infty$ as $\lambda \rightarrow \infty$. The dual function is convex (as a pointwise maximum of affine functions of λ). Projected subgradient descent with diminishing step sizes satisfying $\sum_k \eta_k = \infty$ and $\sum_k \eta_k^2 < \infty$ converges to the optimal set (Bertsekas, 2015, Proposition 3.2.6):

$$\bar{\lambda}_k \rightarrow \lambda^* \in \arg \min_{\lambda \geq 0} d(\lambda).$$

For the case of fixed η (as in the implementation), the iterates converge to a neighborhood of λ^* with radius $O(\eta)$. This is consistent with the standard result for subgradient methods with constant step size (Bertsekas, 2015, Section 3.2.3).

By strong duality (which holds under Assumption (A3); Paternain et al. (2019)), the policy π_{λ^*} is optimal for the original constrained problem (2). $\square$

9.2.4 Function Approximation Extension

Proof of Corollary 4.3. With neural network function approximation, exact convergence guarantees are generally unavailable. However:

Primary and cost heads: Neural network Q-learning seeks stationary points of the projected Bellman error. Practical stability is achieved via Double Q-learning (reduces overestimation), target networks (stabilizes bootstrapping), and experience replay (breaks temporal correlation). With parameter sharing, the effective sample size per network update scales with $\binom{N}{2}$ (all pairwise regions contribute), which improves generalization.

Multiplier update: The dual ascent update (12) with central averaging is a deterministic function of observed collision rates and does not depend on the Q-function parameterization. Its convergence properties are inherited from projected subgradient descent (Bertsekas, 2015), independent of how the Q-functions are represented.

Overall: The algorithm seeks KKT stationary points of the Lagrangian $\mathcal{L}(\theta, \lambda)$, where θ parameterizes the shared Q-network. Convergence to a neighborhood of such points follows from the analysis of nonlinear two-timescale stochastic approximation (Borkar & Borkar, 2008), with the neighborhood size proportional to the approximation error ϵ_{NN} :

$$\text{dist}((\theta_t, \bar{\lambda}_t), \mathcal{K}) \leq O\left(\frac{\epsilon_{\text{NN}}}{1 - \gamma}\right),$$

where $\mathcal{K}$ is the set of KKT points. □

9.3 Proof of Error Decomposition Results

9.3.1 Bellman Operator Properties

Definition 9.3 (Bellman Operators). Define the *joint Bellman operator* $\mathcal{T}$ and *independent regional Bellman operator* $\mathcal{T}_R$ as:

$$(\mathcal{T}Q)(s, a) := r(s, a) + \gamma \sum_{s'} P(s' | s, a) \max_{a'} Q(s', a'), \quad (26)$$

$$(\mathcal{T}_R Q_R)(s^R, a^R) := r^R(s^R, a^R) + \gamma \sum_{s'^R} P(s'^R | s^R, a^R) \max_{a'^R} Q_R(s'^R, a'^R). \quad (27)$$

Additionally, define the *factored Bellman operator* that preserves the additive structure but performs a joint max:

$$(\hat{\mathcal{T}}\hat{Q})(s, a) := \sum_R r^R(s^R, a^R) + \gamma \sum_{s'} P(s' | s, a) \max_{a'} \sum_R \hat{Q}_R(s'^R, a'^R), \quad (28)$$

where $\hat{Q} = \sum_R \hat{Q}_R$ denotes a factored Q-function.

The distinction between $\hat{\mathcal{T}}$ and the collection $\{\mathcal{T}_R\}$ is essential:

- $\hat{\mathcal{T}}$ performs a *joint* max over all agents' actions, accounting for action coupling across overlapping regions. Its fixed point $\hat{Q}^*$ is the best factored Q-function that correctly handles inter-region coordination.
- Each $\mathcal{T}_R$ performs an *independent* max over actions within region R only, ignoring that these actions also appear in other regions. The collection of fixed points $\{Q_R^*\}$ is what our regional Q-learning algorithm computes.

Lemma 9.4 (Contraction). *The operators $\mathcal{T}$, $\hat{\mathcal{T}}$, and each $\mathcal{T}_R$ are γ -contractions in $\|\cdot\|_\infty$.*

Proof. For $\mathcal{T}$ and $\hat{\mathcal{T}}$: the discount factor γ provides contraction, and the max operator is non-expansive (cf. Step 2 of the proof of Theorem 4.1). For $\mathcal{T}_R$: the same argument applied to the regional state-action space. □

9.3.2 Proof of Proposition 4.5: Structural Error Characterization

Proof. We characterize when the structural error β vanishes.

Part (a): Non-overlapping regions.

Suppose the coordination graph $\mathcal{C}_R$ is such that each agent i belongs to *at most one* region (i.e., the regions are disjoint). Then actions in different regions are completely independent: action a^R in region R does not appear in any other region R' . In this case, the joint max decomposes:

$$\max_{a'} \sum_R Q_R(s'^R, a'^R) = \sum_R \max_{a'^R} Q_R(s'^R, a'^R),$$

because each a'^R can be optimized independently. It follows that $\hat{\mathcal{T}}\hat{Q} = \sum_R \mathcal{T}_R Q_R$ for any factored $\hat{Q} = \sum_R Q_R$. Hence the factored Bellman operator coincides with independent regional operators, and the fixed points agree: $\hat{Q}^* = \sum_R Q_R^*$.

Moreover, by an inductive argument on the value iteration $Q_0 = 0$, $Q_{t+1} = \mathcal{T}Q_t$: the reward factorizes (Assumption (A2)), the transitions factorize (Assumption (A1)), and the max decomposes (non-overlapping regions), so each iterate Q_t factorizes exactly. Taking $t \rightarrow \infty$, $Q^* = \sum_R Q_R^*$ and $\beta = 0$.

Part (b): Overlapping regions.

When regions overlap—that is, some agent i belongs to multiple regions—the joint max does *not* decompose in general. To see why, consider agent i belonging to regions R and R' . The action a^i appears in both $Q_R(s'^R, a'^R)$ and $Q_{R'}(s'^{R'}, a'^{R'})$. Optimizing a^i for Q_R alone may differ from optimizing it for $Q_R + Q_{R'}$ jointly. Hence:

$$\max_{a'} \sum_R Q_R(s'^R, a'^R) \leq \sum_R \max_{a'^R} Q_R(s'^R, a'^R),$$

with equality only in special cases (e.g., when the optimal a^i is the same for all regions containing agent i).

For overlapping coordination graphs, the structural error β captures the gap between the joint and independent Bellman operators:

$$\beta = \sup_{s,a} \left| \sum_R \max_{a'^R} Q_R^*(s'^R, a'^R) - \max_{a'} \sum_R Q_R^*(s'^R, a'^R) \right| \frac{\gamma}{1-\gamma}. \quad (29)$$

This is precisely the per-step “coordination gap” between independent and joint action selection, propagated over the infinite horizon. For the pairwise coordination graph with $N \geq 3$ agents, regions overlap (each agent appears in $N - 1$ pairs), so $\beta \geq 0$ in general.

Part (c): When is β small?

Even for overlapping graphs, β may be small when:

- (i) *Weak inter-region coupling:* If the Q-values are dominated by individual agent terms rather than pairwise interactions, then the independent max is a good approximation to the joint max.
- (ii) *Max-Sum compensates:* Although $\beta > 0$ for independent regional Q-learning, the Max-Sum action selection in Layer 2 performs the *joint* max (approximately). This means the coordination error ϵ_{MS} and the structural error β partially compensate: Max-Sum corrects the action coupling that independent Q-learning misses.

In the Simple Spread environment, the pairwise collision penalties are relatively small compared to the individual landmark rewards. Hence, the coupling between overlapping regions is weak, and the structural error β is small in practice—though not exactly zero as previously claimed. $\square$

Remark 9.5 (Relation to Guestrin et al. (2002)). Guestrin et al. (2002) showed that the optimal Q-function for a factored MDP can be *represented* as a sum of regional functions when the coordination graph captures all reward and transition dependencies. However, their result defines the regional components through a *coupled* fixed-point system (the factored Bellman operator $\hat{\mathcal{T}}$), not through *independent* regional Bellman operators $\{\mathcal{T}_R\}$. The independent regional Q-learning used in CG-CMARL is an approximation that avoids the coupling at the cost of structural error β . Max-Sum message passing recovers part of this lost coordination at action selection time. $\triangle$

9.3.3 Proof of Theorem 4.4: Compositional Error Bound

Proof. We decompose the total error using the triangle inequality.

Step 1: Define intermediate Q-functions.

Let:

- Q^* = optimal joint Q-function (fixed point of $\mathcal{T}$)
- $\hat{Q}^* = \sum_R \hat{Q}_R^*$ = optimal factored Q-function (fixed point of $\hat{\mathcal{T}}$, i.e., the best sum-of-regions approximation with joint action selection)
- $\tilde{Q} = \sum_R Q_R^*$ = sum of independently learned regional Q-functions (fixed points of $\{\mathcal{T}_R\}$)
- $Q_{\text{learned}} = \sum_R Q_{\text{learned},R}$ = learned Q-function (with finite samples and function approximation)

Step 2: Apply triangle inequality.

$$\|Q_{\text{learned}} - Q^*\|_{\infty} \leq \underbrace{\|Q_{\text{learned}} - \tilde{Q}\|_{\infty}}_{(a)} + \underbrace{\|\tilde{Q} - \hat{Q}^*\|_{\infty}}_{(b)} + \underbrace{\|\hat{Q}^* - Q^*\|_{\infty}}_{(c)}. \quad (30)$$

Step 3: Bound term (c) — Structural error.

Term (c) measures the approximation error from restricting to factored Q-functions. By the approximate dynamic programming framework (Bertsekas, 2019, Proposition 6.1), if the factored Bellman operator $\hat{\mathcal{T}}$ approximates the joint operator $\mathcal{T}$ with per-step error $\beta := \sup_Q \|\hat{\mathcal{T}}Q - \mathcal{T}Q\|_{\infty}$, then the fixed-point error satisfies:

$$\|\hat{Q}^* - Q^*\|_{\infty} \leq \frac{\beta}{1 - \gamma}.$$

By Proposition 4.5, $\beta = 0$ when regions are non-overlapping, and $\beta \geq 0$ for overlapping graphs such as the pairwise coordination graph with $N \geq 3$.

Step 4: Bound term (b) — Independent learning error.

Term (b) measures the gap between independent regional Q-learning and the coupled factored optimum. This arises because each $\mathcal{T}_R$ performs an independent max over a'^R , while $\hat{\mathcal{T}}$ performs a joint max over a' . The per-step gap is bounded by the “coordination deficit” from independent action selection:

$$\|\tilde{Q} - \hat{Q}^*\|_{\infty} \leq \frac{\epsilon_{\text{MS}}}{1 - \gamma},$$

where ϵ_{MS} is the Max-Sum coordination error, defined as the worst-case gap between the joint optimum $\max_{a'} \sum_R Q_R(s'^R, a'^R)$ and the action selected by Max-Sum (or, in the case of independent learning, the per-region independent maxima). For tree-structured graphs, $\epsilon_{\text{MS}} = 0$ because Max-Sum recovers the exact joint optimum.

Justification: The factored Bellman operator $\hat{\mathcal{T}}$ and the sum of independent operators differ only in the action selection step. The contraction property ensures that a per-step error of ϵ_{MS} in action selection propagates to at most $\epsilon_{\text{MS}}/(1 - \gamma)$ in the fixed-point error, by the standard error propagation bound for approximate value iteration (Bertsekas, 2019, Proposition 6.1).

Step 5: Bound term (a) — Sampling and representation error.

This combines the finite-sample error and function approximation error:

$$\|Q_{\text{learned}} - \tilde{Q}\|_{\infty} \leq \epsilon_{\text{sample}} + \epsilon_{\text{NN}}.$$

The sampling error satisfies $\epsilon_{\text{sample}} = O(1/\sqrt{n})$ by Proposition 9.7 (see Section 9.5). The representation error $\epsilon_{\text{NN}} := \min_{\theta} \max_{R, s^R, a^R} |Q_{R, \theta}(s^R, a^R) - Q_R^*(s^R, a^R)|$ is the best approximation error achievable by the neural network class.

Step 6: Combine.

$$\|Q_{\text{learned}} - Q^*\|_{\infty} \leq \frac{\beta}{1-\gamma} + \frac{\epsilon_{\text{MS}}}{1-\gamma} + O\left(\frac{1}{\sqrt{n}}\right) + \epsilon_{\text{NN}}.$$

□

9.4 Proof of Policy Suboptimality Bound

Lemma 9.6 (Simulation Lemma (Kearns & Singh, 2002, Lemma 2)). *Let $\hat{Q}$ satisfy $\|\hat{Q} - Q^*\|_{\infty} \leq \epsilon$, and let $\hat{\pi}(s) = \arg \max_a \hat{Q}(s, a)$. Then:*

$$V^*(s) - V^{\hat{\pi}}(s) \leq \frac{2\epsilon}{1-\gamma}.$$

Proof of Policy Suboptimality. By Theorem 4.4, the learned Q-function satisfies

$$\|Q_{\text{learned}} - Q^*\|_{\infty} \leq \epsilon := \frac{\beta + \epsilon_{\text{MS}}}{1-\gamma} + O\left(\frac{1}{\sqrt{n}}\right) + \epsilon_{\text{NN}}.$$

Applying Lemma 9.6:

$$V^*(s) - V^{\hat{\pi}}(s) \leq \frac{2\epsilon}{1-\gamma} = \frac{2}{1-\gamma} \left[\frac{\beta + \epsilon_{\text{MS}}}{1-\gamma} + O\left(\frac{1}{\sqrt{n}}\right) + \epsilon_{\text{NN}} \right].$$

□

9.5 Sampling Error Bound

Proposition 9.7 (Q-Learning Sample Complexity (Even-Dar & Mansour, 2003)). *Consider tabular Q-learning with learning rate $\alpha_t = 1/(1 + \text{visits}(s, a))$, run for T steps. With probability at least $1 - \delta$:*

$$\|Q_T - Q^*\|_{\infty} \leq O\left(\frac{R_{\max}}{(1-\gamma)^2} \sqrt{\frac{\log(|\mathcal{S}||\mathcal{A}|T/\delta)}{n_{\min}}}\right),$$

where $n_{\min} = \min_{s,a} N_T(s, a)$ is the minimum visitation count.

Remark 9.8 (Regional Application). For factored Q-learning, each region R runs independent Q-learning on a state-action space of size $|\mathcal{S}^R| \times |\mathcal{A}^R|$. Applying Proposition 9.7 to each region and taking a union bound over $|\mathcal{C}_R|$ regions:

$$\max_R \|Q_{R,T} - Q_R^*\|_{\infty} \leq O\left(\frac{R_{\max}}{(1-\gamma)^2} \sqrt{\frac{\log(|\mathcal{C}_R| \cdot \max_R |\mathcal{S}^R| |\mathcal{A}^R| \cdot T/\delta)}{n_{\min}}}\right)$$

with probability at least $1 - \delta$. With parameter sharing, transitions from all $\binom{N}{2}$ pairwise regions contribute to training a single network, effectively increasing $n_{\min}$ by a factor of $\binom{N}{2}$ relative to training separate networks. $\triangle$

9.6 Constraint Violation Bound

Proposition 9.9 (Finite-Time Constraint Violation). *Under the assumptions of Theorem 4.2, the cumulative constraint violation over K episodes satisfies*

$$\sum_{k=0}^{K-1} [c_{\text{thresh}} - \bar{c}_k]_+ \leq O\left(\frac{(\lambda_0 - \lambda^*)^2}{\eta} + \eta K\right),$$

where $\bar{c}_k := \frac{1}{N} \sum_i c_k^i$ is the team-average collision cost in episode k . Optimizing over η yields a cumulative violation of $O(\sqrt{K})$.

Proof Sketch. This follows from the regret analysis of projected subgradient descent on the dual function. The dual update moves $\bar{\lambda}$ toward λ^* at rate η , with the per-episode constraint violation serving as the subgradient. The standard bound for projected subgradient descent with constant step size (Bertsekas, 2015, Section 3.2.3) yields the result. The $O(\sqrt{K})$ rate is achieved by choosing $\eta \propto 1/\sqrt{K}$, which balances the bias-variance tradeoff. $\square$

10 Implementation and Experimental Details

This appendix describes the Simple Spread environment used in our experiments (Section 10.1), the reward and constraint decompositions (Section 10.2), computational complexity (Section 10.3), and a discussion of when the factorization introduces error (Section 10.4). For hyperparameter values and network architecture, see Appendix 10.

10.1 Simple Spread Environment

The `simple_spread` environment from PettingZoo/MPE (Lowe et al., 2017) consists of N agents and N landmarks in a 2D continuous space. Agents must cover all landmarks (each landmark should have at least one nearby agent) while avoiding inter-agent collisions.

State Space. Each agent i observes its own velocity and position $(\text{vel}_i, \text{pos}_i) \in \mathbb{R}^4$, the positions of all landmarks $(\text{pos}_{\ell_1}, \dots, \text{pos}_{\ell_N}) \in \mathbb{R}^{2N}$, and the positions of all other agents $(\text{pos}_k)_{k \neq i} \in \mathbb{R}^{2(N-1)}$. The global state is the concatenation of all agent states.

Action Space. Each agent applies a continuous 2D force. We discretize each dimension into 5 bins (including no-op), yielding $|\mathcal{A}^i| = 25$ actions per agent. For a pairwise region $R = \{i, k\}$, the joint action space has $|\mathcal{A}^R| = 625$ elements.

Verification of Assumptions. Simple Spread satisfies both structural assumptions:

- (A1) *Independent transitions.* Each agent’s position and velocity update depends only on its own state and action (applied force). There are no physical rebounds on collision—agents pass through each other, incurring only a reward penalty.
- (A2) *Additive reward.* The global reward decomposes into individual landmark-distance terms and pairwise collision penalties (see Section 10.2).

Scaling Experiments. We evaluate CG-CMARL with $N \in \{3, 4, 6, 10\}$ agents. The observation dimension, number of pairwise regions, and joint action space grow as shown in Table 11.

Table 11: Scaling of problem dimensions with N agents.

N	$ C_R $	$ o^R $	$ \mathcal{A} ^N$	Status
3	3	16	1.6×10^4	Tractable (brute force)
4	6	18	3.9×10^5	Tractable (brute force)
6	15	22	2.4×10^8	Intractable centrally
10	45	30	9.5×10^{13}	Intractable centrally

10.2 Reward and Constraint Decomposition

10.2.1 Primary Reward: Per-Agent Counterfactual Rewards

The global coverage utility is the negative sum of minimum agent-to-landmark distances:

$$U(\mathcal{V}) = - \sum_{\ell=1}^N \min_{i \in \mathcal{V}} \|\text{pos}_i - \text{pos}_\ell\|.$$

For each pairwise region $R = \{i, k\}$, the primary reward is the *per-agent counterfactual reward* (Wolpert & Tumer, 2001):

$$r_0^R = \sum_{i \in R} [U(\mathcal{V}) - U(\mathcal{V} \setminus \{i\})], \quad (31)$$

which sums the marginal contributions of each agent in R to global coverage. Compared to the global counterfactual $U(\mathcal{V}) - U(\mathcal{V} \setminus R)$, which removes the entire region at once, the per-agent variant distinguishes the contribution of each agent within the region.

Relation to Assumption (A2). For non-overlapping coordination graphs (e.g., the pairwise graph with $N = 2$), the per-agent counterfactual rewards satisfy the additive factorization (1) exactly: each agent belongs to a single region, so marginal contributions do not overlap (Proposition 4.5(i)). For overlapping pairwise graphs ($N \geq 3$), the marginal contributions overlap and $\sum_R r_0^R \neq r_0$ in general. We use per-agent counterfactual rewards in this regime following standard practice in cooperative MARL (Wolpert & Tumer, 2001), given their strong credit assignment properties: each agent is rewarded only for the coverage improvement it individually produces. The resulting discrepancy is absorbed into the structural error β , which the compositional bound (Theorem 4.4) already characterizes. The `local_ratio` parameter (set to 0.0) controls the mixture between per-agent counterfactual rewards and local rewards; at 0.0, the primary reward contains no collision penalty.

10.2.2 Collision Cost

In the implementation, collision costs $c^R \geq 0$ are computed directly; the constraint reward used in the theoretical formulation (Sections 3.1–3.3) is $r_1^R = -c^R$, so that the augmented Q-function (5) with $+\lambda Q_{\text{cost}}$ penalizes collisions.

For the cost head, each pairwise region $R = \{i, k\}$ observes a binary collision indicator:

$$c^R(s^R) = \mathbf{1}[\|\text{pos}_i - \text{pos}_k\| < d_{\text{thresh}}], \quad (32)$$

where $d_{\text{thresh}} = 0.2$ is the collision distance. The cost head learns $Q_{\text{cost}, R}$, the expected discounted collision frequency for the pair (i, k) .

10.2.3 Constraint Definition

The Lagrangian multiplier penalizes excessive collisions. Agent i 's episode-level collision cost is

$$c_{\text{episode}}^i = \frac{1}{H} \sum_{t=1}^H \sum_{k \neq i} \mathbf{1}[\|\text{pos}_i(t) - \text{pos}_k(t)\| < d_{\text{thresh}}],$$

where H is the episode length. In terms of the constraint reward convention, $\hat{V}_1^i = -c_{\text{episode}}^i$, so the dual update (12) increases λ^i when $c_{\text{episode}}^i > c_{\text{thresh}}$ (equivalently, when $\hat{V}_1^i < -c_{\text{thresh}}$) and decreases it otherwise. With $c_{\text{thresh}} = 0$, the constraint aims for zero collisions; in practice, the clamp to $[0, \lambda_{\text{max}}]$ limits the multiplier's influence, allowing the Pareto sweep to explore the full coverage–safety tradeoff.

10.3 Computational Complexity

Table 12 compares the per-step computational cost of centralized, factored (CG-CMARL), and fully independent approaches.

Table 12: Computational complexity comparison. Here $|A|$ denotes the per-agent action size, $|R|_{\text{max}}$ the largest region, and K the number of Max-Sum iterations.

Method	Action Selection	Q-Update
Centralized	$O(A ^N)$	$O(A ^N)$ per sample
Factored (CG-CMARL)	$O(\mathcal{C}_R A ^{ R _{\text{max}}} K)$	$O(\mathcal{C}_R A ^{ R _{\text{max}}})$ per sample
Independent (IQL)	$O(N A)$	$O(N A)$ per sample

For the pairwise graph ($|R|_{\text{max}} = 2$, $|\mathcal{C}_R| = \binom{N}{2}$) with $|A| = 25$ and $K = 10$:

Table 13: Concrete action-selection cost for Simple Spread.

N	Centralized	CG-CMARL	IQL
3	1.6×10^4	1.9×10^4	75
4	3.9×10^5	3.8×10^4	100
6	2.4×10^8	9.4×10^4	150
10	9.5×10^{13}	2.8×10^5	250

Interpretation. For $N = 3$, CG-CMARL is comparable in cost to centralized search. The advantage becomes dramatic for $N \geq 6$, where centralized action selection is intractable while CG-CMARL remains polynomial. The cost relative to IQL is the price of coordination: CG-CMARL's $O(N^2)$ scaling reflects the pairwise interaction structure, while IQL's $O(N)$ ignores it entirely. With parameter sharing, the Q-update cost is independent of $|\mathcal{C}_R|$ since all regions share a single network—the $|\mathcal{C}_R|$ factor applies only to transition collection and Max-Sum message passing.

10.4 When Factorization Introduces Error

The structural error β (Proposition 4.5) arises from two sources, which we discuss in turn.

Source 1: Action Overlap in Pairwise Graphs. As shown in Proposition 4.5(ii), for pairwise coordination graphs with $N \geq 3$ agents, the independent per-region max does not equal the joint max over all agents' actions. This contributes $\beta > 0$ even when Assumptions (A1)–(A2) hold perfectly. The magnitude of this contribution depends on the strength of coupling between overlapping regions—specifically, how much the optimal action for agent i in region R_{ij} differs from its optimal action considering all regions $\{R_{ik}\}_{k \neq i}$ simultaneously. In Simple Spread, this coupling is driven by collision penalties, which are typically small relative to landmark rewards.

Source 2: Transition Dependencies. If the environment violates Assumption (A1)—for instance, if collisions cause physical rebounds or agents draw from shared resources—then the regional Bellman operator $\mathcal{T}_R$ (27) is no longer exact, because $P(s'^R | s, a) \neq P(s'^R | s^R, a^R)$. This adds a transition-coupling component to β .

Source 3: Reward Dependencies Beyond the Graph. If the reward contains interactions not captured by the coordination graph (e.g., three-body interactions with only pairwise regions), then Assumption (A2) is violated and β includes a reward-coverage component.

Example 10.1 (Higher-Order Reward Interaction). In a cooperative defense scenario where a target’s state depends on simultaneous attacks from three agents:

$$r_{\text{capture}}(s, a^1, a^2, a^3) \neq \sum_{i < j} r^{ij}(s^{\{i,j\}}, a^{\{i,j\}}),$$

a pairwise graph cannot represent this interaction. Extending $\mathcal{C}_R$ to include triplet regions captures the dependency at the cost of increasing the per-region action space from $|A|^2$ to $|A|^3$.

Practical Guidance. The choice of coordination graph involves a tractability–accuracy tradeoff:

Richer graphs (larger regions, more hyperedges): smaller β , but larger per-region action spaces and denser factor graphs (slower Max-Sum, more parameters).

Sparser graphs (pairwise or tree-structured): larger β but faster action selection and simpler message passing (K can be smaller; $\epsilon_{\text{MS}} = 0$ for trees).

The compositional error bound (Theorem 4.4) provides a principled way to reason about this tradeoff: increasing graph richness reduces β but may increase ϵ_{MS} (denser cycles) and ϵ_{NN} (larger input/output dimensions).

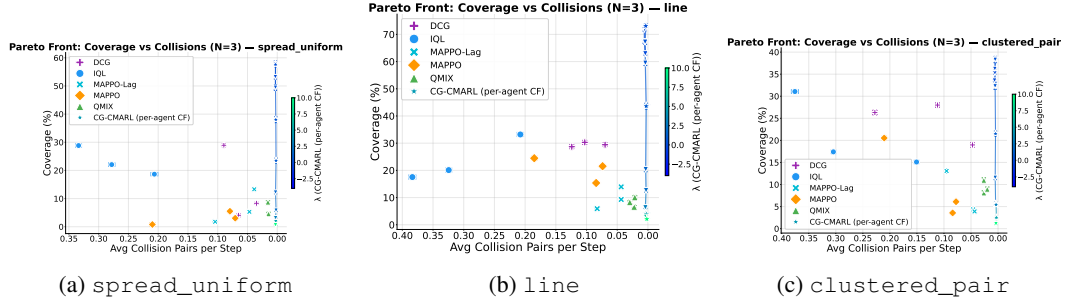
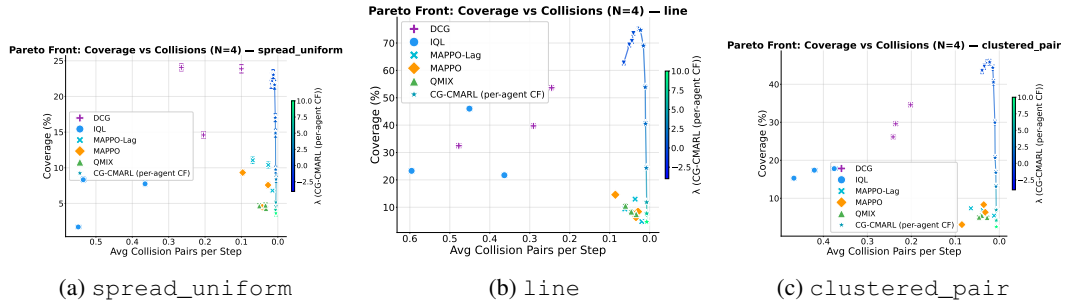
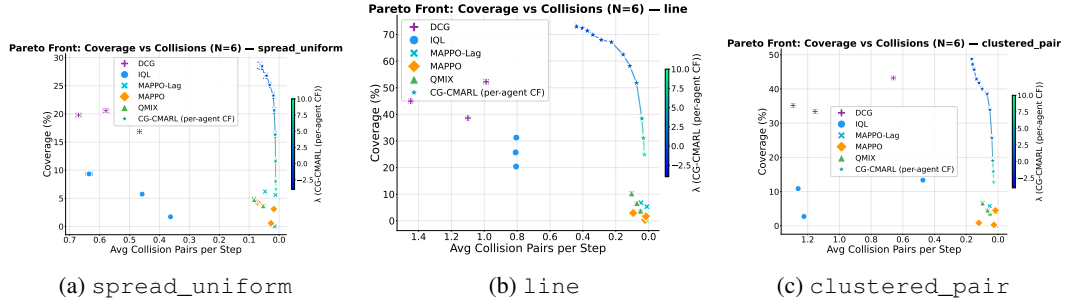
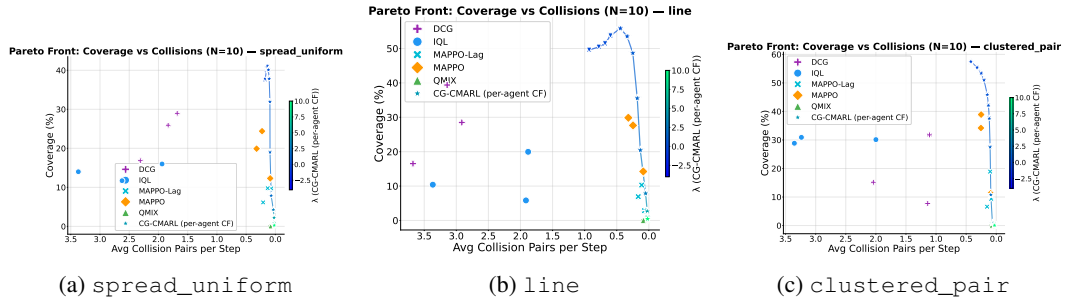
10.5 Pareto Evaluation on Fixed Landmark Scenarios

The Pareto fronts in Section 5.2 are evaluated on environments with randomized initial landmark positions. To further probe how each method behaves under controlled layouts, we additionally evaluate every trained model (CG-CMARL and all baselines) on three fixed landmark scenarios from our scenario registry (Appendix 10.1):

- `spread_uniform`: N landmarks placed evenly on a circle of radius 0.7 (regular polygon vertices). This is a balanced layout with no geometric bias.
- `line`: N landmarks evenly spaced along a horizontal corridor at $y = 0$. Tests whether agents can split along a linear structure rather than a 2D cluster.
- `clustered_pair`: two clusters of roughly $N/2$ landmarks, one centred at $(-0.5, -0.5)$ and one at $(0.5, 0.5)$. Tests whether agents split into groups and commit to distant subsets.

Training is unchanged (agents train on random scenarios); only evaluation is run on these fixed layouts. For each scenario we evaluate the same trained models used in Section 5.2 at the same λ sweep for CG-CMARL and at the same reward-shaping settings for the baselines. Each operating point is averaged over multiple evaluation episodes with fixed landmarks but randomized initial agent positions.

Figures 2–5 show the resulting Pareto fronts, with rows indexed by team size $N \in \{3, 4, 6, 10\}$ and columns by scenario (`spread_uniform`, `line`, `clustered_pair`). Each panel plots the average number of colliding agent pairs per step (x-axis, inverted; lower is better) against coverage (y-axis), with CG-CMARL’s λ -sweep tracing a curve and each baseline appearing as a single operating point. The qualitative pattern observed in the random-scenario evaluation persists across fixed layouts: CG-CMARL’s front traces or dominates the baselines, particularly in the moderate- λ regime. Performance on `clustered_pair` is the most sensitive to team size, reflecting the difficulty of splitting agents between distant subsets as N grows.


 Figure 2: Pareto fronts on fixed landmark scenarios for $N = 3$.

 Figure 3: Pareto fronts on fixed landmark scenarios for $N = 4$.

 Figure 4: Pareto fronts on fixed landmark scenarios for $N = 6$.

 Figure 5: Pareto fronts on fixed landmark scenarios for $N = 10$.

10.6 Safety Metric and Fairness Across Team Sizes

The safety metric reported throughout the paper is the *average number of overlapping agent pairs per step*: at each timestep we count unordered pairs (i, k) whose Euclidean distance is below $2r_{\text{agent}}$, then average over steps and episodes. This quantity is exactly the cost signal the Lagrangian multiplier penalizes during training (the cost head predicts it per region), so reporting it as the evaluation metric keeps the training objective and the evaluation objective aligned.

The metric has one property that is worth making explicit: its maximum possible value is $\binom{N}{2}$, which grows quadratically in the team size. A raw value of 0.05 collisions per step therefore carries different weight at $N = 3$ (where the maximum is 3) than at $N = 10$ (where the maximum is 45). This means cross- N comparisons using the raw value understate how well the algorithm scales.

To disentangle “total safety burden on the system” from “per-pair safety,” Tables 15–16 report both the raw metric and a normalized *per-pair* rate defined as

$$\text{per-pair rate} := \frac{\text{Collisions}}{\binom{N}{2}}. \quad (33)$$

The per-pair rate admits a clean interpretation: it is the probability that a uniformly-random pair of agents is in collision at a uniformly-random timestep.

Empirical scaling of CG-CMARL. Table 14 compares CG-CMARL at the same multiplier ($\lambda = 1$) across team sizes. In raw units the collision count grows by roughly $15\times$ from $N = 3$ to $N = 10$, suggesting that safety degrades sharply with team size. The per-pair rate tells a very different story: it stays essentially flat, between 0.0011 and 0.0019. That is, CG-CMARL maintains a nearly constant per-pair collision probability as the team scales; the apparent degradation in the raw metric is almost entirely a combinatorial artifact of there being more pairs to track.

Table 14: CG-CMARL at $\lambda = 1$: raw collisions vs. per-pair rate as team size grows. The raw metric’s $\approx 15\times$ increase reflects the quadratic growth in $\binom{N}{2}$, not a genuine degradation of per-pair safety.

N	$\binom{N}{2}$	Collisions	Per-pair rate
3	3	0.0034	0.00113
4	6	0.0103	0.00172
6	15	0.0289	0.00193
10	45	0.0534	0.00119

We keep the raw pair count as the primary metric because it matches the training objective, but the per-pair rate is the more appropriate quantity for cross- N comparisons and is reported alongside in the tables below.

10.7 Full Numerical Results

Tables 15–16 report the full numerical results summarized by the Pareto front plots in Section 5.2. Collisions denotes the average number of colliding agent pairs per step (lower is better), matching the x-axis of the Pareto plots; coverage is reported as a percentage. All values are averaged over 2 random seeds; $\pm$ denotes one sample standard deviation. For CG-CMARL, ★ marks Pareto-optimal operating points. These tables are auto-generated by `scripts/generate_appendix_tables.py` directly from the per-seed `pareto_sweep.json` files.

Table 15: Baseline results across all team sizes. “lr” = local_ratio; “cl” = cost_limit (for MAPPO-Lagrangian). Collisions is the average number of colliding agent pairs per step (lower is better); per-pair rate is Collisions divided by $\binom{N}{2}$ and allows fair comparison across N (see Appendix 10.6); coverage is reported as a percentage.

Algorithm	Param.	Collisions	Per-pair	Coverage (%)
<i>N = 3 (3 pairwise regions)</i>				
IQL	lr=0.0	0.0519 ± 0.0081	0.0173 ± 0.0027	20.58 ± 0.82
IQL	lr=0.3	0.0623 ± 0.0075	0.0208 ± 0.0025	17.42 ± 3.42
IQL	lr=0.6	0.0483 ± 0.0001	0.0161 ± 0.0000	15.67 ± 0.47
QMIX	lr=0.0	0.0052 ± 0.0008	0.0017 ± 0.0003	7.83 ± 0.42
QMIX	lr=0.3	0.0060 ± 0.0005	0.0020 ± 0.0002	5.60 ± 0.66
QMIX	lr=0.6	0.0071 ± 0.0009	0.0024 ± 0.0003	5.83 ± 0.24
DCG	lr=0.0	0.0450 ± 0.0040	0.0150 ± 0.0013	21.58 ± 3.42
DCG	lr=0.3	0.0439 ± 0.0081	0.0146 ± 0.0027	23.50 ± 1.89
DCG	lr=0.6	0.0322 ± 0.0003	0.0107 ± 0.0001	24.17 ± 2.36
MAPPO	lr=0.0	0.0232 ± 0.0240	0.0077 ± 0.0080	9.84 ± 5.33
MAPPO	lr=0.3	0.0126 ± 0.0003	0.0042 ± 0.0001	8.00 ± 2.36
MAPPO	lr=0.6	0.0084 ± 0.0058	0.0028 ± 0.0019	10.90 ± 3.64
MAPPO-Lag	cl=0.1	0.0185 ± 0.0047	0.0062 ± 0.0016	10.53 ± 1.51
MAPPO-Lag	cl=1.0	0.0067 ± 0.0012	0.0022 ± 0.0004	9.97 ± 0.33
<i>N = 4 (6 pairwise regions)</i>				
IQL	lr=0.0	0.0837 ± 0.0100	0.0140 ± 0.0017	17.19 ± 0.97
IQL	lr=0.3	0.0868 ± 0.0011	0.0145 ± 0.0002	16.00 ± 1.94
IQL	lr=0.6	0.0703 ± 0.0047	0.0117 ± 0.0008	16.56 ± 1.68
QMIX	lr=0.0	0.0182 ± 0.0005	0.0030 ± 0.0001	3.70 ± 0.35
QMIX	lr=0.3	0.0177 ± 0.0008	0.0030 ± 0.0001	3.95 ± 0.14
QMIX	lr=0.6	0.0187 ± 0.0021	0.0031 ± 0.0004	4.72 ± 0.25
DCG	lr=0.0	0.0856 ± 0.0181	0.0143 ± 0.0030	21.50 ± 4.42
DCG	lr=0.3	0.0858 ± 0.0051	0.0143 ± 0.0008	24.69 ± 1.33
DCG	lr=0.6	0.0756 ± 0.0020	0.0126 ± 0.0003	23.56 ± 0.62
MAPPO	lr=0.0	0.0161 ± 0.0096	0.0027 ± 0.0016	10.61 ± 3.79
MAPPO	lr=0.3	0.0171 ± 0.0123	0.0029 ± 0.0021	5.19 ± 2.21
MAPPO	lr=0.6	0.0172 ± 0.0183	0.0029 ± 0.0031	9.88 ± 2.47
MAPPO-Lag	cl=0.1	0.0122 ± 0.0032	0.0020 ± 0.0005	7.97 ± 0.18
MAPPO-Lag	cl=1.0	0.0139 ± 0.0119	0.0023 ± 0.0020	11.18 ± 2.65
<i>N = 6 (15 pairwise regions)</i>				
IQL	lr=0.0	0.1058 ± 0.0060	0.0071 ± 0.0004	15.83 ± 2.36
IQL	lr=0.3	0.0960 ± 0.0141	0.0064 ± 0.0009	14.21 ± 0.53
IQL	lr=0.6	0.0802 ± 0.0032	0.0053 ± 0.0002	12.87 ± 0.06
QMIX	lr=0.0	0.0217 ± 0.0021	0.0014 ± 0.0001	1.85 ± 0.07
QMIX	lr=0.3	0.0243 ± 0.0021	0.0016 ± 0.0001	1.80 ± 0.14
QMIX	lr=0.6	0.0313 ± 0.0013	0.0021 ± 0.0001	2.88 ± 0.21
DCG	lr=0.0	0.1425 ± 0.0193	0.0095 ± 0.0013	24.17 ± 0.24
DCG	lr=0.3	0.1458 ± 0.0051	0.0097 ± 0.0003	24.00 ± 0.47
DCG	lr=0.6	0.1544 ± 0.0025	0.0103 ± 0.0002	24.17 ± 1.65
MAPPO	lr=0.0	0.0107 ± 0.0020	0.0007 ± 0.0001	2.82 ± 3.65
MAPPO	lr=0.3	0.0174 ± 0.0044	0.0012 ± 0.0003	2.88 ± 2.77
MAPPO	lr=0.6	0.0183 ± 0.0078	0.0012 ± 0.0005	7.23 ± 3.96
MAPPO-Lag	cl=0.1	0.0188 ± 0.0083	0.0013 ± 0.0006	4.48 ± 3.13
MAPPO-Lag	cl=1.0	0.0310 ± 0.0055	0.0021 ± 0.0004	7.37 ± 3.63
<i>N = 10 (45 pairwise regions)</i>				
IQL	lr=0.0	0.2087 ± 0.0010	0.0046 ± 0.0000	16.73 ± 1.87

(continued on next page)

(continued from previous page)

Algorithm	Param.	Collisions	Per-pair	Coverage (%)
IQL	lr=0.3	0.2035 ± 0.0068	0.0045 ± 0.0002	15.80 ± 0.14
IQL	lr=0.6	0.1759 ± 0.0029	0.0039 ± 0.0001	15.12 ± 2.09
QMIX	lr=0.0	0.0365 ± 0.0090	0.0008 ± 0.0002	0.86 ± 1.10
QMIX	lr=0.3	0.0193 ± 0.0017	0.0004 ± 0.0000	0.00 ± 0.00
QMIX	lr=0.6	0.0222 ± 0.0038	0.0005 ± 0.0001	0.00 ± 0.00
DCG	lr=0.0	0.2272 ± 0.0083	0.0050 ± 0.0002	22.43 ± 1.17
DCG	lr=0.3	0.2263 ± 0.0015	0.0050 ± 0.0000	21.02 ± 0.95
DCG	lr=0.6	0.2145 ± 0.0054	0.0048 ± 0.0001	19.28 ± 0.25
MAPPO	lr=0.0	0.0798 ± 0.0408	0.0018 ± 0.0009	17.03 ± 3.21
MAPPO	lr=0.6	0.0387 ± 0.0239	0.0009 ± 0.0005	17.40 ± 4.27
MAPPO-Lag	cl=0.5	0.0247 ± 0.0070	0.0005 ± 0.0002	12.59 ± 12.26

Table 16: CG-CMARL results across all team sizes. Each row corresponds to a single trained model evaluated at the indicated λ . Collisions is the average number of colliding agent pairs per step (lower is better); per-pair rate is Collisions divided by $\binom{N}{2}$ for fair cross- N comparison (see Appendix 10.6). ★ = Pareto-optimal operating point.

N	λ	Collisions	Per-pair	Coverage (%)	Pareto
3	0.00	0.0055 ± 0.0001	0.0018 ± 0.0000	47.58 ± 2.24	★
	0.05	0.0054 ± 0.0003	0.0018 ± 0.0001	46.58 ± 6.01	
	0.10	0.0052 ± 0.0006	0.0017 ± 0.0002	47.33 ± 2.59	★
	0.20	0.0037 ± 0.0004	0.0012 ± 0.0001	46.17 ± 0.94	★
	0.50	0.0042 ± 0.0006	0.0014 ± 0.0002	38.08 ± 0.12	
	1.00	0.0034 ± 0.0003	0.0011 ± 0.0001	17.50 ± 2.83	★
	2.00	0.0030 ± 0.0014	0.0010 ± 0.0005	8.25 ± 1.53	★
	5.00	0.0017 ± 0.0001	0.0006 ± 0.0000	3.92 ± 2.24	★
	10.00	0.0022 ± 0.0006	0.0007 ± 0.0002	1.33 ± 1.18	
4	0.00	0.0243 ± 0.0016	0.0041 ± 0.0003	37.50 ± 0.88	
	0.05	0.0186 ± 0.0034	0.0031 ± 0.0006	38.94 ± 1.86	★
	0.10	0.0171 ± 0.0027	0.0029 ± 0.0004	36.56 ± 2.03	
	0.20	0.0137 ± 0.0007	0.0023 ± 0.0001	38.50 ± 0.53	★
	0.50	0.0162 ± 0.0006	0.0027 ± 0.0001	32.06 ± 3.45	
	1.00	0.0103 ± 0.0021	0.0017 ± 0.0004	26.81 ± 1.33	★
	2.00	0.0070 ± 0.0011	0.0012 ± 0.0002	16.50 ± 3.89	★
	5.00	0.0075 ± 0.0013	0.0013 ± 0.0002	6.75 ± 0.53	
	10.00	0.0064 ± 0.0011	0.0011 ± 0.0002	4.38 ± 1.24	★
6	0.00	0.0413 ± 0.0022	0.0028 ± 0.0001	33.29 ± 3.01	
	0.05	0.0457 ± 0.0006	0.0030 ± 0.0000	33.75 ± 1.65	
	0.10	0.0390 ± 0.0012	0.0026 ± 0.0001	34.83 ± 0.24	★
	0.20	0.0428 ± 0.0021	0.0029 ± 0.0001	33.92 ± 1.89	
	0.50	0.0299 ± 0.0020	0.0020 ± 0.0001	33.21 ± 0.77	★
	1.00	0.0289 ± 0.0015	0.0019 ± 0.0001	28.67 ± 0.35	★
	2.00	0.0172 ± 0.0016	0.0011 ± 0.0001	22.38 ± 2.89	★
	5.00	0.0153 ± 0.0012	0.0010 ± 0.0001	11.75 ± 1.30	★
	10.00	0.0120 ± 0.0002	0.0008 ± 0.0000	7.33 ± 0.94	★
10	0.00	0.1223 ± 0.0177	0.0027 ± 0.0004	27.14 ± 1.47	
	0.10	0.1167 ± 0.0130	0.0026 ± 0.0003	27.18 ± 0.03	
	0.20	0.1074 ± 0.0124	0.0024 ± 0.0003	27.64 ± 1.41	★
	0.50	0.0815 ± 0.0144	0.0018 ± 0.0003	24.69 ± 0.18	★
	1.00	0.0534 ± 0.0114	0.0012 ± 0.0003	21.80 ± 0.71	★
	2.00	0.0343 ± 0.0035	0.0008 ± 0.0001	12.24 ± 1.73	★
	5.00	0.0198 ± 0.0010	0.0004 ± 0.0000	1.85 ± 0.41	★
	10.00	0.0157 ± 0.0002	0.0003 ± 0.0000	0.52 ± 0.14	★