Population Contracts for Scalable Language-Guided Multi-Robot Planning and Control

Anonymous submission

Abstract

Large heterogeneous robot teams must translate natural-language missions into coordinated behavior without making planning and control representations depend on a fixed list of robot identities. Existing language-model planners often enumerate robots, while centralized multi-agent reinforcement learning commonly concatenates agent states; both choices complicate transfer to new team sizes. We propose a shared population-level interface between language planning, task allocation, and decentralized control. A *Population Contract* specifies group goals, roles, proportional quotas, capability requirements, priorities, constraints, and recovery conditions without naming robots. A validator and capability-aware optimizer ground the contract to available agents, after which a role-conditioned population policy maps local observations, assigned contract entries, capabilities, and group statistics to continuous actions. We establish conditional grounding soundness, identity-equivariance of the actor, identity-invariance of the critic, and population-independent network dimensions for a fixed schema. The implemented stack integrates contract validation, CP-SAT grounding, VMAS, BenchMARL training, and deterministic exact-population evaluation. In the current $N = 10$ coverage slice, MAPPO and the initial concatenation-based Group-MF both complete 500k-frame training, but the latter achieves only 0.12/0.05 static/dynamic success versus 0.21/0.09 for random control. All nine subsequent fixed-data actor runs pass a pre-registered imitation-and-rollout gate, and a behavior-cloning warm start raises seed-0 gated-residual Group-MF success to 50/50 episodes. Thus, the present evidence validates the cross-layer implementation and localizes the immediate bottleneck to on-policy discovery; it does not yet establish language-cost scaling or unseen-population transfer.

Introduction

Natural-language interfaces could make heterogeneous robot teams easier to task in logistics, inspection, and disaster response. The difficulty is not only interpreting a mission, but also turning it into assignments and continuous actions for a team whose composition may change during deployment. Recent large language model (LLM) systems decompose instructions, form coalitions, and revise plans for multiple robots (Kannan, Venkatesh, and Min 2024; Nayak et al. 2024; Chen et al. 2025). However, prompts and dialogue histories that enumerate robots grow with team size. Empirical comparisons of centralized, decentralized, and hybrid LLM planners likewise show that communication structure materially affects token use and coordination success (Chen et al. 2024).

Moving to team-level planning only partially resolves this issue. PIP-LLM generates robot-independent Planning Domain Definition Language (PDDL) actions and subsequently uses integer programming for capability-aware assignment (Shi et al. 2025). This separation is effective for structured task planning, but assumes a static closed world and does not define how a learned continuous controller should consume or revise the high-level plan. Conversely, scalable multi-agent reinforcement learning (MARL) compresses variable populations through mean fields, sets, or graphs (Yang et al. 2018; Nayak et al. 2023; Zaheer et al. 2017), yet typically receives goals that are already assigned and exposes no interface to a language planner. Capability-aware policies transfer across heterogeneous team compositions, but explicitly leave high-level planning and allocation outside their scope (Howell et al. 2023).

We study whether these layers can share one population-level semantic representation. The proposed *Population Contract* contains a compact set of group goals, roles, proportional quotas, capability requirements, priorities, constraints, and termination or replanning conditions. Its entries are indexed by semantic groups rather than robot identities. Only a capability-aware allocator binds entries to concrete robots; both the LLM and the learned controller operate on group-level quantities. This distinction is important: permutation invariance removes dependence on identity order, but does not imply that an optimal policy can ignore population size (Li et al. 2023). We therefore retain normalized counts, density, and resource-to-population ratios while avoiding fixed-length identity lists.

The framework closes the loop through hierarchical recovery. Local deviations are handled by the decentralized policy, agent failures or quota shortfalls trigger reallocation, and the LLM is called only when the contract becomes infeasible or mission semantics change. This decomposition tests a single research question: can a shared identity-decoupled

This is an anonymized submission for review purposes only. Distribution, citation, or public sharing of this manuscript is strictly prohibited. Copyright and publication details will appear in the final version if accepted.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

contract reduce the growth of language-planning cost while preserving task success, satisfying constraints, and transferring zero-shot to unseen populations and dynamic missions?

This paper makes three contributions:

- It defines a machine-checkable Population Contract as a shared interface among language planning, capability-aware allocation, decentralized control, and execution feedback.
- It formulates verified contract grounding and a role-conditioned population actor–critic, and establishes their conditional grounding, relabeling, and representation-size properties.
- It implements and audits the contract–allocator–control path, reports a deterministic $N = 10$ pilot, and uses fixed-data and PPO diagnostics to distinguish representational capacity from failed on-policy behavior discovery.

Related Work

Language-Based Multi-Robot Planning

LLM-based systems have decomposed household instructions and assigned subtasks to small robot teams (Kannan, Venkatesh, and Min 2024), closed long-horizon execution loops with correction and verification (Nayak et al. 2024), and represented heterogeneous embodiments through robot resumes (Chen et al. 2025). These approaches usually expose robot-specific observations, histories, or actions to one or more LLMs. Hybrid communication reduces some of this cost but still reasons over robot-level actions (Chen et al. 2024). Structured approaches instead translate language to formal planning problems (Liu et al. 2023) or robot-independent team actions followed by optimized assignment (Shi et al. 2025). Affordance grounding further shows that linguistic relevance must be combined with executable skills (Ahn et al. 2022). Event-driven systems such as CoMuRoS support plan revision, but maintain robot-level states and local LLM executors (Borate et al. 2025). Our focus is not another task-decomposition prompt; it is the representation passed consistently from decomposition through continuous control.

Scalable and Heterogeneous MARL

MAPPO is a strong cooperative MARL baseline, but a critic formed by concatenating all agent states is tied to the training population (Yu et al. 2022). Mean-field MARL replaces many-agent interactions with a neighborhood average (Yang et al. 2018), while InforMARL uses agent–entity graphs and global mean pooling to support changing numbers of entities (Nayak et al. 2023). Mean-field MAPPO has also been instantiated for large competitive teams using population distributions in the critic (Jeloka, Guan, and Tsiotras 2025). Deep Sets and Set Transformers provide alternative permutation-invariant representations with different relational and computational trade-offs (Zaheer et al. 2017; Zhang et al. 2022). Population-size-aware policy optimization demonstrates that invariant policies may still need explicit size information (Li et al. 2023). For heterogeneity, continuous capability inputs improve transfer to unseen robots and compositions (Howell et al. 2023), and capability-conditioned hypernetworks specialize shared policies (Fu et al. 2025). Role learning can also induce behavioral specialization (Wang et al. 2020). These methods motivate our controller, but none uses the same explicit role and quota specification as a language planner and allocator.

Research Gap

Language planning, task allocation, and variable-population control each have scalable components, but their interfaces remain semantically disconnected. Team-level symbolic plans do not directly condition continuous policies, and population-invariant policies do not tell a language planner which group statistics and constraints will remain meaningful after grounding. We address this gap by making a validated Population Contract the common object manipulated by all three layers.

Problem Formulation

Consider a family of heterogeneous decentralized partially observable Markov decision processes indexed by team size N. The agent set is $\mathcal{A}_N = \{1, \dots, N\}$. Agent i has capability vector $c_i \in \mathbb{R}^{d_c}$, receives local observation o_i^t, and selects continuous action a_i^t. The team receives reward r_t and safety or resource costs $d_t^{(m)}$, each constrained by budget $D^{(m)}$, following a constrained Markov game formulation (Gu et al. 2023). A natural-language mission u_t and an aggregated environment description $\bar{s}_t$ may change during an episode.

The high-level planner produces a Population Contract

$$\mathcal{P}_t = \{(g_k, r_k, \rho_k, [\ell_k, u_k], q_k, p_k, C_k, \tau_k)\}_{k=1}^{K_t}. \quad (1)$$

Here g_k and r_k are a group goal and role; ρ_k is a proportional quota; $[\ell_k, u_k]$ is an admissible group-size interval; q_k is a capability requirement; p_k is priority; C_k contains safety, temporal, or mutual-exclusion constraints; and τ_k defines completion and replanning conditions. The number of entries K_t is determined by mission semantics, not by N.

An allocator maps $(\mathcal{P}_t, \{c_i, o_i^t\}_{i=1}^N)$ to assignments $z_i^t \in \{0, \dots, K_t\}$, where zero denotes unassigned reserve. A shared policy π_θ maps local information and the assigned contract entry to actions. We evaluate: (i) mission success and constrained costs; (ii) zero-shot performance when test N is absent from training; and (iii) the slopes of contract length, LLM tokens and latency, allocation time, policy step time, and memory with respect to N.

Population Contract Framework

Overview

Figure 1 shows the planning–allocation–control loop. The LLM sees a mission and aggregated environment/capability statistics, then emits a structured contract. A validator checks syntax, semantics, and allocation feasibility. The allocator is the only component that binds robot identities. Assigned agents execute a shared role-conditioned policy, and group-level execution summaries select among local recovery, reallocation, and LLM replanning.

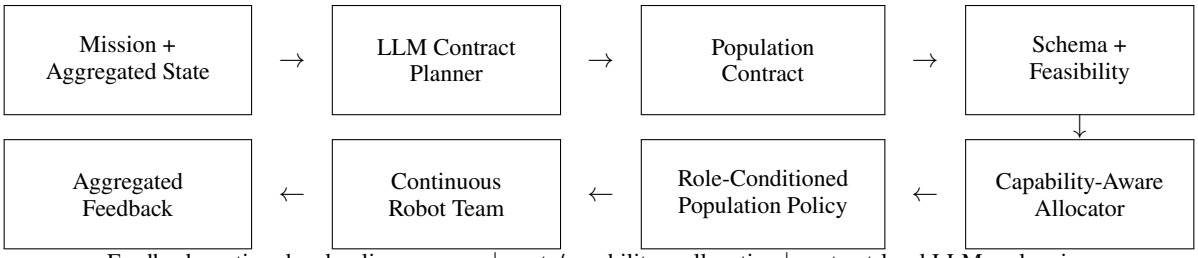

Figure 1: The Population Contract is the shared group-level interface. Its size follows the number of semantic groups K, while only the allocator enumerates the N available robots. Execution feedback is aggregated by contract entry and escalated according to the scope of failure.

Contract Generation and Validation

Aggregated input. Robot-level prompts grow even when additional robots are interchangeable. We instead summarize capabilities by type-conditioned counts, normalized histograms or moments, and availability; spatial state is summarized by semantic region, goal, obstacle density, and resource demand. The summary retains N, density, and resource-to-population ratios, because removing identities should not remove population context. It does not claim constant size with respect to arbitrary mission complexity: input and contract length may grow with semantic regions, object classes, and groups.

Structured generation. The planner emits Equation 1 under a fixed schema rather than free-form robot actions. This follows the broader principle of using an LLM to construct a formal problem that an external solver can verify (Liu et al. 2023; Shi et al. 2025). Absolute group size is derived from the proportional quota:

$$n_k = \min(u_k, \max(\ell_k, \mathrm{round}(\rho_k N))) \,. \tag{2}$$

Thus, a contract can be reused when N changes, while explicit bounds prevent unsafe under- or over-staffing.

Validation. Validation proceeds at three levels. Syntactic validation checks types, required fields, normalized quotas, and references among entries. Semantic validation rejects contradictions such as a completion condition that violates a hard constraint. Feasibility validation invokes the allocator without committing assignments. On failure, the validator returns a compact conflict set—for example, insufficient payload capability for a high-priority transport group—rather than a list of failed robots. This feedback grounds language generation in executable capability, analogous to affordance-based skill filtering (Ahn et al. 2022).

Capability-Aware Contract Grounding

Let $x_{ik} \in \{0, 1\}$ indicate whether robot i is assigned to entry k, and let M_{ik} encode individual and non-additive compatibility. For additive capability requirements, A_k selects or transforms the relevant capability dimensions. The allocator enforces

$$\sum_{k=1}^{K} x_{ik} \le 1, \qquad \forall i, \tag{3}$$

$$\ell_k \le \sum_{i=1}^{N} x_{ik} \le u_k, \qquad \forall k, \tag{4}$$

$$x_{ik} \le M_{ik}, \qquad \forall i, k, \tag{5}$$

$$\sum_{i=1}^{N} x_{ik} A_k c_i \succeq b_k, \qquad \forall k. \tag{6}$$

Equation 3 permits a reserve pool, Equation 4 enforces population quotas, Equation 5 excludes robots lacking mandatory skills, and Equation 6 requires the assigned group to cover aggregate capability demand b_k. Temporal precedence, colocation, and mutual exclusion in C_k add standard linear constraints. This is a time-extended multi-task, multi-robot allocation problem in the taxonomy of Gerkey and Mataric (2004).

The current CP-SAT implementation minimizes a weighted sum of travel and reassignment costs subject to Equations 3–6; the second term discourages identity churn after local failures. A priority-ordered greedy allocator provides a diagnostic baseline. The solver can return feasible incumbents under a runtime budget, while an independent verifier rechecks every committed assignment and a group-level explanation reports capability or quota shortages when no solution is found. The design separates semantic group creation from combinatorial grounding: the LLM does not search over $N \times K$ assignments.

Role-Conditioned Population Policy

Local population representation. A single global mean obscures heterogeneous roles. Let $\mathcal{R}$ be a fixed role vocabulary and $\mathcal{N}_{i,r}^t$ the neighbors of agent i observed in role $r \in \mathcal{R}$. We compute a normalized role-conditioned embedding

$$\mu_{i,r}^t = \frac{1}{\max(1, |\mathcal{N}_{i,r}^t|)} \sum_{j \in \mathcal{N}_{i,r}^t} \phi_\theta(o_j^t, c_j, e_{r_j}), \tag{7}$$

where ϕ_θ is a shared encoder and e_{r_j} is the role embedding. An empty-role mask distinguishes no observed members

from a zero-valued mean. Compared with a single neighborhood average (Yang et al. 2018), Equation 7 preserves coarse multimodality across roles while retaining fixed dimensionality for a bounded role vocabulary.

Decentralized actor. The shared actor receives

$$\begin{aligned} q_i^t &= \left[o_i^t, c_i, e(g_{z_i}), e(r_{z_i}), \{\mu_{i,r}^t\}_{r\in\mathcal{R}}\right], \\ h_i^t &= \left[q_i^t, \hat{\rho}^t, N/N_{\max}, \delta_i^t\right], \qquad a_i^t \sim \pi_\theta(\cdot \mid h_i^t), \end{aligned} \tag{8}$$

where $\hat{\rho}^t$ is the realized role distribution and δ_i^t is local density. Goal, role, and capability conditioning permit agents sharing parameters to specialize without identity embeddings, consistent with capability-aware transfer (Howell et al. 2023; Fu et al. 2025) and role-conditioned control (Wang et al. 2020). Explicit size and density variables address the fact that an invariant optimum can still depend on finite population size (Li et al. 2023).

Population critic. During centralized training, each contract entry produces

$$\begin{aligned} \bar{h}_k^t &= \frac{1}{\max(1, n_k)} \sum_{i:z_i=k} \phi_\omega(o_i^t, c_i), \\ v_k^t &= \psi_\omega\big(e(g_k), e(r_k), \rho_k, \bar{h}_k^t, \xi_k^t\big), \end{aligned} \tag{9}$$

where ξ_k^t contains progress and constraint statistics. The critic pools the set of group embeddings,

$$\begin{aligned} \alpha_k^t &= \frac{\exp(w^\top v_k^t)}{\sum_{j=1}^K \exp(w^\top v_j^t)}, \\ V_\omega(s_t, \mathcal{P}_t) &= \rho_\omega\left(\sum_{k=1}^K \alpha_k^t v_k^t\right). \end{aligned} \tag{10}$$

The sum and attention weights are permutation invariant over contract entries (Zaheer et al. 2017). Unlike joint-state concatenation, the critic architecture is unchanged when N changes. Set Transformer and graph critics remain important baselines because richer relations may outperform compressed means (Zhang et al. 2022; Nayak et al. 2023).

Training objective. We optimize the clipped MAPPO surrogate

$$\begin{aligned} \bar{r}_t(\theta) &= \mathrm{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon), \\ \mathcal{L}_{\mathrm{actor}}(\theta) &= -\mathbb{E}\Big[\min\Big(r_t(\theta)\hat{A}_t, \bar{r}_t(\theta)\hat{A}_t\Big)\Big] - \beta\mathcal{H}(\pi_\theta). \end{aligned} \tag{11}$$

with value loss $\mathcal{L}_V(\omega) = \mathbb{E}[(V_\omega - \hat{R}_t)^2]$. Here $r_t(\theta)$ is the policy likelihood ratio, $\hat{A}_t$ the advantage estimate, ϵ the clipping threshold, β the entropy coefficient, and $\hat{R}_t$ the return target. The registered constrained-control comparison augments this objective with separate cost critics and nonnegative Lagrange multipliers for $D^{(m)}$, yielding a MAPPO-Lagrangian baseline (Gu et al. 2023).

Event-Triggered Hierarchical Recovery

At interval H, execution is summarized for each entry as

$$f_k^t = (p_k^t, n_k^t, m_k^t, v_k^t, s_k^t), \tag{12}$$

representing progress, active quota, capability margin, violations, and stagnation, respectively. The summary contains no trajectories or robot dialogue. Failures are handled at the lowest sufficient level: (1) transient tracking and collision-avoidance deviations remain within the policy; (2) persistent quota or capability loss invokes the allocator while holding $\mathcal{P}_t$ fixed; and (3) infeasibility, priority changes, new goals, or sustained stagnation invokes the LLM. This differs from calling a corrector at every high-level step (Nayak et al. 2024) and from assigning a local LLM executor to each robot (Borate et al. 2025).

Scaling Properties

The framework separates three notions of scalability. The high-level representation contains K entries and is $O(K)$ in semantic groups, although its aggregate statistics require $O(N)$ preprocessing. Grounding has $O(NK)$ binary variables and remains a combinatorial optimization problem; population abstraction does not eliminate allocation complexity. For a neighborhood graph with E directed edges, policy aggregation and the centralized critic require $O(N + E + K)$ work; the current dense radius-graph backend constructs edges in $O(N^2)$, while a cell-list backend avoids all-pairs construction when occupancy is bounded. Experiments must measure all three costs instead of inferring runtime scalability from permutation invariance alone.

Formal Properties

The following statements concern the finite contract and neural representations, not language-model correctness, controller optimality, or polynomial-time grounding.

Proposition 1 (Verified grounding soundness)**.** *If the allocator returns an assignment that is accepted by the independent verifier, that assignment satisfies uniqueness, resolved group sizes, individual compatibility, and additive payload and energy requirements encoded by Equations 3–6.*

Proof. The verifier reconstructs the robot–group incidence matrix from the returned identifiers and directly checks each encoded constraint using the unrounded robot capabilities and resolved quotas. Acceptance is therefore a certificate for the modeled constraints. This guarantee is conditional on the contract schema and does not cover requirements omitted or mistranslated by the language model. □

Proposition 2 (Identity relabeling)**.** *Let σ permute the active-agent dimension while carrying each agent's observation, capability, role, group, and position together. Then the shared actor is equivariant, $\pi_\theta(\sigma O) = \sigma\pi_\theta(O)$, and the population critic is invariant, $V_\omega(\sigma O, \mathcal{P}) = V_\omega(O, \mathcal{P})$.*

Proof. Pairwise distances and role memberships are preserved by relabeling. Each role-conditioned neighborhood is aggregated by a mean, so the corresponding context is unchanged except for the destination-agent order. The shared actor consequently permutes its outputs with its inputs. The critic first sums agents within anonymous groups and then applies a weighted sum over group embeddings; both operations are invariant to the order of their operands. □

Proposition 3 (Population-independent dimensions)**.** *For fixed observation features, role vocabulary $|\mathcal{R}|$, and maximum contract width $K_{\max}$, the contract encoding, per-agent actor input, actor parameters, and critic parameters do not grow with N. Runtime and memory need not be constant: aggregation processes N agents and E neighborhood edges, and grounding contains $O(NK)$ binary variables.*

Proof. All learned maps consume fixed-width agent features, $|\mathcal{R}|$ role summaries, or at most $K_{\max}$ pooled group embeddings. No parameter is indexed by robot identity. The stated runtime terms follow from visiting agents and edges and from constructing one assignment variable per robot–group pair. □

Experiments

Evidence Scope

We separate formal claim tests from engineering and diagnostic evidence. The implemented slice contains the contract parser, three-stage validator, CP-SAT and greedy allocators, multi-region coverage, an OpenAI-compatible structured planner adapter, the role-conditioned actor and population critic, checkpointed BenchMARL training, and exact-population evaluation. The current quantitative results cover only fixed-area coverage at exact $N = 10$, with one trained seed for reinforcement-learning policies. They can test software integrity and expose optimization failures, but cannot establish language-cost scaling, cross-task generality, multi-seed performance, or zero-shot transfer.

Registered Formal Evaluation

The complete registered suite contains capability-constrained transport, multi-region coverage, and dynamic rescue in VMAS (Bettini et al. 2022), trained through BenchMARL (Bettini, Prorok, and Moens 2024). Policies train on $N \in \{10, 30, 50\}$ and are evaluated without updates at $N = 100$, on both fixed-area and density-controlled maps. The planning comparison uses individual-level, hybrid, PIP-style, and Population-Contract prompts with the same `gpt-4.1-2025-04-14` model, information, decoding, structured-output, and repair budgets. The control comparison includes MAPPO (Yu et al. 2022), vanilla mean-field MAPPO, Deep Sets, Set Transformer, an InforMARL-style graph critic (Nayak et al. 2023), and capability-aware policies (Howell et al. 2023; Fu et al. 2025). The allocation comparison includes random feasible, nearest compatible, greedy capability matching, and CP-SAT.

The formal study uses five independently trained seeds and 200 paired episodes per task, population, map regime, and event condition. Its primary metrics are success, completion, quota satisfaction, cost per robot-step, $N = 100$ retention, learning-curve area, tokens, calls, latency, allocation time, policy time, throughput, and peak memory. Binary outcomes use paired exact tests; continuous outcomes use paired bootstrap intervals and Wilcoxon tests with Holm correction. The $N = 100$ set is excluded from model selection. These rules describe the evidence required for the central research question; the following pilot is reported separately because it does not meet that protocol.

500k Training and Frozen-Policy Pilot

Training integrity. On an RTX 4090, MAPPO and Group-MF each completed 500,000 frames in 100 collection–optimization iterations with no recorded non-finite value or runtime failure. MAPPO required 52.50 minutes at 158.72 frames/s and 726.23 MiB peak CUDA memory; Group-MF required 51.63 minutes at 161.40 frames/s and 1,047.78 MiB. Both final checkpoints passed structural validation, SHA-256 verification, and full BenchMARL reload. The single sequential run therefore supports execution and persistence, not a throughput advantage.

Exact-population evaluation. We migrated each trained actor strictly to an exact-$N = 10$ environment and evaluated 100 paired episodes in static coverage and 100 with a midpoint sensing/priority event. The dynamic path performs CP-SAT reallocation, updates contract and assignment versions, and records assignment churn. All 600 method–condition episodes completed; repeated deterministic subsets matched, and every dynamic episode reached contract version 2 and assignment version 2 with mean churn 6.

Table 1 rejects a positive claim for the initial concatenation-based Group-MF at this stage. MAPPO exceeds random success by 0.28 in static and 0.31 in dynamic coverage, but its cost is approximately 18 times the random value. Group-MF is 0.09 below random in static success and 0.04 below it dynamically; it is also about 50% slower per policy step than MAPPO at $N = 10$. Because the actor training seed is singular and the pilot reward lacks the registered cost critic, these differences are descriptive and must not be generalized to algorithms.

Controlled Optimization Diagnostics

Task solvability and actor capacity. After replacing sparse-context concatenation with gated residual fusion and adding distinct role goals and local collision lidar, a deterministic goal-directed controller solved 20/20 episodes in 38.45 mean steps with cost 0.0421 per robot-step. Nevertheless, three controlled 100k-frame from-scratch iterations failed their continuation gate: the best Group-MF success was 0.04, versus 0.24 for paired random control. We therefore froze the environment and fitted local MAPPO, global-mean, and Group-MF actors to disjoint oracle train/validation/test trajectories.

All 9/9 runs passed the preregistered MSE, success, cost, and checkpoint-reload thresholds in Table 2. The result shows that the implemented Group-MF actor can represent the controller on the oracle distribution; it does not show that role-conditioned context is superior, because all representations solve the target.

On-policy numerical probes. A strictly paired one-iteration PPO probe used byte-identical CPU-collected trajectories and CUDA optimization for three methods and three seeds. All 18 condition runs were finite, all nine pairs matched initial parameters and collection hashes, and both actor and critic updated. Advantage normalization increased the actor-gradient norm by 56–78 times, yet the normalized/current deterministic action-change ratios ranged from 0.43 to 1.40 and averaged only 0.92–0.94 across methods. It

Table 1: Single-seed exact-$N = 10$ frozen-policy pilot. Each row contains 100 paired episodes. Higher success and quota are better; lower steps, cost per robot-step, and policy-step time are better. Bold and underline mark the best and second-best value within each condition. Cost was logged but was not optimized by a constrained objective in this pilot.

Condition	Policy	Success ↑	Quota ↑	Steps ↓	Cost/step ↓	Policy ms ↓
Static	Random	<u>0.21</u>	<u>0.4200</u>	<u>445.77</u>	**0.0243**	**0.0167**
Static	MAPPO	**0.49**	**0.8150**	**290.59**	0.4435	<u>0.0710</u>
Static	Group-MF (concat)	0.12	0.4175	452.24	<u>0.1836</u>	0.1064
Dynamic	Random	<u>0.09</u>	0.3425	493.38	**0.0242**	**0.0169**
Dynamic	MAPPO	**0.40**	**0.7725**	**402.92**	0.4378	<u>0.0708</u>
Dynamic	Group-MF (concat)	0.05	<u>0.3975</u>	<u>490.67</u>	<u>0.1910</u>	0.1074

Table 2: Fixed-data actor diagnostic over three initialization seeds. MSE and cost are seed means; rollout success aggregates 20 held-out episodes per seed. The Group-MF actor uses gated residual fusion. Bold and underline mark the best and second-best value. These are imitation diagnostics, not RL comparisons.

Actor	Params	Test MSE ↓	Success	Cost ↓
Local MLP	24,450	**0.000432**	60/60	**0.05632**
Global mean	28,739	<u>0.000433</u>	60/60	0.05639
Group-MF	49,539	0.000441	60/60	<u>0.05634</u>

therefore did not justify a production change. Reducing only the initial action scale lowered Group-MF cost from 0.4248 to 0.1929 but reduced success from 2/50 to 0/50, so that hypothesis also failed.

Warm-start diagnosis. Mapping a successful Group-MF behavior-cloning checkpoint into the PPO actor produced 20/20 success before and after one real 5k-frame PPO iteration; cost changed from 0.03117 to 0.03063 while actor and critic relative parameter changes were nonzero. This indicates that one PPO update preserves a successful policy and shifts the immediate diagnosis toward discovery from random trajectories. In the final controlled gate, changing only the actor initialization to the fixed behavior-cloning checkpoint yielded 50/50 success after 100k frames, 44.9 mean steps, and cost 0.05131; the paired random policy obtained 12/50 success and cost 0.02202. This establishes a runnable demonstration-initialized loop, but not a fair method comparison: formal use requires the same demonstration budget for every learned baseline and a from-scratch ablation.

Current Claim Boundary

The available evidence supports the finite contract, verified grounding, representation symmetries, exact-N execution, deterministic recovery plumbing, and the diagnosis that the present random-initialized Group-MF training does not learn the coverage controller. It does not support the central claims that Population Contracts reduce token-growth slope, preserve success and safety across tasks, or improve zero-shot $N = 100$ retention. Those claims require successful authenticated LLM trials, transport and rescue environments, the full set/graph/capability baselines, five-seed training, matched demonstrations if warm starts are retained, registered ablations, and profiling over $N = 10, 30, 50, 100$.

Limitations

The abstraction has clear limits. Role-conditioned means can discard topology needed for tightly coupled maneuvers, contract length can grow with mission complexity, and constrained grounding remains combinatorial. The guarantees in Propositions 1–3 are conditional on a correct finite schema and do not validate natural-language semantics or establish policy optimality. The current dense neighborhood construction is quadratic in N, and the Python cell-list implementation requires device-level profiling before it can support a runtime advantage claim. VMAS abstracts low-level dynamics, and evaluation up to 100 robots cannot establish physical-robot or thousand-agent scalability. More immediately, the reported reinforcement-learning pilot has one training seed, one task, one population, and no constrained cost objective. The behavior-cloning warm start changes the information budget and is not comparable to from-scratch baselines until demonstrations are matched. The language adapter has not completed an authenticated online generation, so no token, latency, or semantic-validity result is reported.

Conclusion

We presented Population Contracts as a shared group-level interface for language-guided planning, capability-aware assignment, and decentralized multi-robot control. The formulation confines identity binding to the allocator while giving planning and control common goals, roles, quotas, capabilities, constraints, and recovery conditions. The structural results and execution gates establish verified grounding, relabeling behavior, fixed learned dimensions, checkpointed training, and exact-population recovery. The current pilot also provides a negative result: random-initialized Group-MF does not outperform random control on $N = 10$ coverage. Fixed-data and retention probes show that the actor can represent the target behavior and that PPO can preserve it, while behavior-cloning initialization closes the minimal control loop. These findings narrow the optimization problem but do not answer the central scalability question. Claims about lower language cost, safety preservation, and unseen-population transfer remain contingent on the registered multi-task, multi-seed, and $N = 100$ experiments.

References

Ahn, M.; et al. 2022. Do As I Can, Not As I Say: Grounding Language in Robotic Affordances. In *Conference on Robot Learning*.

Bettini, M.; Kortvelesy, R.; Blumenkamp, J.; and Prorok, A. 2022. VMAS: A Vectorized Multi-Agent Simulator for Collective Robot Learning. arXiv:2207.03530.

Bettini, M.; Prorok, A.; and Moens, V. 2024. BenchMARL: Benchmarking Multi-Agent Reinforcement Learning. *Journal of Machine Learning Research*, 25(217): 1–10.

Borate, S.; Rai B, B.; Pardeshi, V.; and Vadali, M. 2025. CoMuRoS: A LLM-Driven Hierarchical Architecture for Adaptive Multi-Robot Collaboration. arXiv:2511.22354.

Chen, J.; et al. 2025. EMOS: Embodiment-Aware Heterogeneous Multi-Robot Operating System with LLM Agents. In *International Conference on Learning Representations*.

Chen, Y.; Arkin, J.; Zhang, Y.; Roy, N.; and Fan, C. 2024. Scalable Multi-Robot Collaboration with Large Language Models: Centralized or Decentralized Systems? In *IEEE International Conference on Robotics and Automation*.

Fu, K.; Jain, S.; Howell, P.; and Ravichandar, H. 2025. Capability-Aware Shared Hypernetworks for Flexible Heterogeneous Multi-Robot Coordination. In *Conference on Robot Learning*.

Gerkey, B. P.; and Mataric, M. J. 2004. A Formal Analysis and Taxonomy of Task Allocation in Multi-Robot Systems. *International Journal of Robotics Research*, 23(9): 939–954.

Gu, S.; Kuba, J. G.; Wen, M.; Chen, R.; Wang, Z.; Tian, Z.; Wang, J.; Knoll, A.; and Yang, Y. 2023. Multi-Agent Constrained Policy Optimisation. In *Conference on Uncertainty in Artificial Intelligence*.

Howell, P.; Rudolph, M.; Torbati, R.; Fu, K.; and Ravichandar, H. 2023. Generalization of Heterogeneous Multi-Robot Policies via Awareness and Communication of Capabilities. In *Conference on Robot Learning*, 2772–2790.

Jeloka, B.; Guan, Y.; and Tsiotras, P. 2025. Learning Large-Scale Competitive Team Behaviors with Mean-Field Interactions. arXiv:2504.21164.

Kannan, S. S.; Venkatesh, V. L. N.; and Min, B.-C. 2024. SMART-LLM: Smart Multi-Agent Robot Task Planning Using Large Language Models. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 12140–12147.

Li, P.; et al. 2023. Population-Size-Aware Policy Optimization for Mean-Field Games. In *International Conference on Learning Representations*.

Liu, B.; Jiang, Y.; Zhang, X.; Liu, Q.; Zhang, S.; Biswas, J.; and Stone, P. 2023. LLM+P: Empowering Large Language Models with Optimal Planning Proficiency. arXiv:2304.11477.

Nayak, S.; Choi, K.; Ding, W.; Dolan, S.; Gopalswamy, S.; and Balakrishnan, H. 2023. Scalable Multi-Agent Reinforcement Learning through Intelligent Information Aggregation. In *International Conference on Machine Learning*, 25817–25833.

Nayak, S.; et al. 2024. LLaMAR: Long-Horizon Planning for Multi-Agent Robots in Partially Observable Environments. In *Advances in Neural Information Processing Systems*, volume 37.

Shi, G.; et al. 2025. PIP-LLM: Integrating PDDL-Integer Programming with LLMs for Coordinating Multi-Robot Teams Using Natural Language. arXiv:2510.22784.

Wang, T.; Dong, H.; Lesser, V.; and Zhang, C. 2020. ROMA: Multi-Agent Reinforcement Learning with Emergent Roles. In *International Conference on Machine Learning*, 9876–9886.

Yang, Y.; Luo, R.; Li, M.; Zhou, M.; Zhang, W.; and Wang, J. 2018. Mean Field Multi-Agent Reinforcement Learning. In *International Conference on Machine Learning*, 5571–5580.

Yu, C.; Velu, A.; Vinitsky, E.; Gao, Y.; Wang, Y.; Bayen, A.; and Wu, Y. 2022. The Surprising Effectiveness of PPO in Cooperative Multi-Agent Games. In *Advances in Neural Information Processing Systems*, volume 35.

Zaheer, M.; Kottur, S.; Ravanbakhsh, S.; Poczos, B.; Salakhutdinov, R.; and Smola, A. J. 2017. Deep Sets. In *Advances in Neural Information Processing Systems*, volume 30.

Zhang, F.; et al. 2022. Relational Reasoning via Set Transformers: Provable Efficiency and Applications to MARL. In *Advances in Neural Information Processing Systems*, volume 35.