Coordination Matters: Evaluation of Cooperative Multi-Agent Reinforcement Learning

Maria Ana Cardei *, **Matthew Landers, Afsaneh Doryab**
University of Virginia

Abstract

Cooperative multi-agent reinforcement learning (MARL) benchmarks commonly emphasize aggregate outcomes such as return, success rate, or completion time. While essential, these metrics often fail to reveal how agents coordinate, particularly in settings where agents, tasks, and joint assignment choices scale combinatorially. We propose a coordination-aware evaluation perspective that supplements return with process-level diagnostics. We instantiate this perspective using STAT, a controlled commitment-constrained spatial task-allocation testbed that systematically varies agents, tasks, and environment size while holding observation access and task rules fixed. We evaluate six representative value-based MARL methods across varying levels of centralization. Our results show that similar return trends can reflect distinct coordination mechanisms, including differences in redundant assignment, assignment diversity, and task-completion efficiency. We find that in commitment-constrained task allocation, performance under scale is shaped not only by nominal action-space size, but also by assignment pressure, sparse decision opportunities, and redundant choices among interdependent agents. Our findings motivate coordination-aware evaluation as a necessary complement to return-based benchmarking for cooperative MARL.[2]

1 Introduction

Cooperative multi-agent reinforcement learning (MARL) studies settings in which multiple agents learn to act in a shared environment to optimize a common objective [5, 24]. In such systems, performance depends not only on individual agents' skills, but also on their ability to coordinate. Agents must avoid redundant behavior, divide work effectively, and adapt to the actions of others. As the number of agents, tasks, and available decisions grows, coordination becomes increasingly difficult because the joint action space can scale combinatorially with these factors [23, 14]. In these settings, aggregate reward alone may be insufficient to explain why a multi-agent system succeeds or fails.

Most empirical evaluations of cooperative MARL rely primarily on outcome-level measures such as return, success rate, or completion time to evaluate methods [19, 28, 27, 20, 33]. These metrics are essential for measuring task performance, but they provide limited visibility into the coordination process that produces that performance. Two policies may obtain similar return while relying on different interaction patterns, and conversely, a change in return may conflate poor coordination, inefficient division of labor, under-utilization of agents, or domain-specific bottlenecks. This limitation is especially important for benchmarks intended to evaluate scaling behavior, where increasing the number of agents, tasks, or available choices may change not only task difficulty but also the structure of coordination itself. This motivates a coordination-aware evaluation perspective, in which coopera-

*cbr8ru@virginia.edu
[2]Code is available at `https://github.com/mariacardei/coordination_aware_MARL`.

Preprint.

tive MARL benchmarks report process-level diagnostics that characterize how agents coordinate, in addition to their achieved return.

Existing cooperative MARL benchmarks have driven substantial progress by standardizing algorithm evaluation, including StarCraft micromanagement [28], particle-world coordination tasks [19], level-based foraging [9], and Overcooked-style collaboration [8]. These benchmarks capture challenges such as partial observability, communication, credit assignment, and collaborative planning. However, there remains a need for controlled testbeds that isolate how coordination changes under systematic combinatorial scaling, where the number of agents, tasks, and available joint actions increase while task rules and observation access remain fixed.

We study this issue through commitment-constrained spatial task allocation, a problem class with roots in multi-robot task allocation, spatially distributed planning, and spatial crowdsourcing [13, 10, 3, 35]. In this setting, agents assign themselves to spatially distributed tasks and commit to completing them over time. This induces structured combinatorial coordination, as assignment choices interact across agents, and commitment makes the effective action space state-dependent. To instantiate this evaluation perspective, we use STAT (the Spatial Task Allocation Testbed), a configurable cooperative MARL testbed that scales agents, tasks, and environment size under full observability. STAT uses action masking and finite-state commitment to isolate high-level task-allocation coordination. We leverage STAT to compare MARL methods across varying training and execution centralization regimes and report coordination-aware process-level diagnostics tailored to this setting, including total task assignment conflicts, conflict rate, conflicts per task, task completion throughput, and per-agent task assignment diversity. These metrics reveal coordination failure modes that return alone can conceal (Figure 1), highlighting that ***in addition to measuring the return, cooperative MARL benchmarks should also assess how well agents coordinate with each other.***

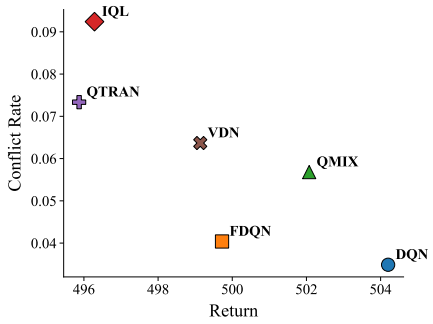


Figure 1: Conflict rate provides a complementary diagnostic beyond return.

Our contributions are as follows:

- We motivate coordination-aware evaluation for cooperative MARL under combinatorial scaling, emphasizing process-level diagnostics beyond return.
- We instantiate this evaluation perspective in STAT, a controlled commitment-constrained spatial task-allocation testbed that supports systematic scaling over agents, tasks, and environment size while holding observation access and task rules fixed.
- We define task-allocation-specific process diagnostics that capture redundant assignment, allocation quality, and task-completion efficiency, including conflict rate, conflicts per task, per-agent assignment diversity, and task throughput.
- We provide an empirical comparison of MARL methods across varying levels of training and execution centralization regimes, showing that return can obscure distinct coordination failure modes as agents, tasks, and spatial scale increase.

2 Related Work

Coordination-Aware Evaluation. Prior work has recognized that aggregate performance does not fully characterize multi-agent coordination. The broader multi-agent systems literature has proposed coordination-specific measures for complex team behavior [21], while MARL work has studied behavioral diagnostics such as role diversity [15] and agent-level coordination measures [37]. Related work in human-AI and human-team collaboration similarly shows that high reward or team score do not imply effective cooperation, motivating interaction-level measures such as constructive interdependence, collaborative actions, and division of labor [4, 30]. While existing work motivates evaluation protocols that measure coordination processes directly, rather than relying solely on aggregate task outcomes, it does not focus on controlled benchmark evaluation under systematic combinatorial scaling. We address this gap through spatial task allocation, a natural setting for coordination-aware evaluation because agents must distribute themselves across shared tasks,

making redundant assignments and poor workload distribution directly measurable. We analyze how these process metrics change as agents, tasks, and spatial extent are independently scaled.

Spatial Task Allocation Settings. Spatial task allocation has been studied across multi-agent systems and robotics. This setting studies teams of agents servicing spatially distributed tasks, where centralized planning becomes difficult as the number of agents and tasks grows [10]. Other formulations consider dynamic spatial and temporal constraints, including soft deadlines and sequential execution requirements [3]. Related spatial crowdsourcing work studies analogous worker–task matching problems under geographic constraints, often focusing on geographic partitioning, heterogeneous spatial data, or platform-mediated assignment [35, 17, 38, 11]. Recent robotics and warehouse work has also applied reinforcement learning to task allocation, including attention-based policies for multi-robot warehouse task allocation and approaches that jointly address task allocation and navigation [1, 2]. While existing work motivates spatial task allocation as an important coordination problem, it generally focuses on allocation algorithms, crowdsourcing objectives, or integrated task-allocation and navigation systems. In contrast, we use spatial task allocation as a controlled cooperative MARL setting for coordination-aware evaluation, with process-level diagnostics that make assignment failures directly observable.

Cooperative MARL Benchmarks. Existing cooperative MARL benchmarks have enabled standardized evaluation across domains such as StarCraft micromanagement [28], particle-world coordination [19], level-based foraging [9], Overcooked-style collaboration [8], and warehouse coordination [26]. These environments capture important challenges such as partial observability, communication, credit assignment, collaborative planning, navigation, and domain-specific interaction dynamics. However, their richness can also make coordination difficult to diagnose, as performance differences may reflect coordination quality, but may also be affected by observability constraints, navigation bottlenecks, sparse rewards, congestion, object manipulation, or domain-specific mechanics. We use STAT as a controlled instantiation of this evaluation gap. It abstracts spatial task allocation into a domain-general setting in which agents, tasks, and environment size can be systematically scaled while task rules and observation access remain fixed. By isolating assignment coordination as the dominant failure mode and making redundant assignments observable through process metrics, STAT enables controlled analysis of coordination behavior beyond return. Table 2 in Appendix B compares STAT with relevant cooperative MARL benchmarks.

3 Preliminaries

We model a cooperative MARL problem with n agents as a fully cooperative Markov game [18]. It is defined by the tuple $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, P, R, \gamma \rangle$, where $\mathcal{S}$ is the global state space, $\mathcal{A}$ is the joint action space, $P(s_{t+1} \mid s_t, a_t)$ is the transition function, $R(s_t, a_t)$ is the shared reward function, and $\gamma \in [0, 1)$ is the discount factor. At each time step t, the environment is in state $s_t \in \mathcal{S}$, each agent i selects an action $a_t^i \in \mathcal{A}_i$, and the resulting joint action is denoted by $a_t = (a_t^1, \ldots, a_t^n) \in \mathcal{A}$. The objective is to find a joint policy $\pi = (\pi_1, \ldots, \pi_n)$ that maximizes the expected discounted return,

$$J(\pi) = \mathbb{E}_{\pi,P}\left[\sum_{t=0}^{T} \gamma^t R(s_t, a_t)\right].$$

Equivalently, the optimal joint policy is given by $\pi^* = \arg\max_\pi J(\pi)$. The joint action space is formed as the Cartesian product of individual agent action spaces: $\mathcal{A}_{\text{joint}} = \mathcal{A}_1 \times \mathcal{A}_2 \times \cdots \times \mathcal{A}_n$, so that each joint action $a_t \in \mathcal{A}$ is a structured combination of individual agent actions. This induces a combinatorial action space whose size grows exponentially with the number of agents: $|\mathcal{A}_{\text{joint}}| = \prod_{i=1}^{n} |\mathcal{A}_i|$. If all agents share the same action space size, i.e., $|\mathcal{A}_i| = |\mathcal{A}_{\text{local}}|$ for all i, then this simplifies to $|\mathcal{A}| = |\mathcal{A}_{\text{local}}|^n$. This combinatorial growth makes learning and coordination increasingly difficult, particularly when the value of one agent's action depends strongly on the simultaneous actions of others.

4 Coordination-Aware Evaluation Design

We instantiate coordination-aware evaluation in a controlled commitment-constrained spatial task-allocation setting. The design has three goals: (1) expose structured combinatorial scaling through

agents, tasks, and environment size, (2) isolate task-assignment coordination as the primary coordination bottleneck by holding observation access fixed and abstracting away low-level collision avoidance and path planning, and (3) support process-level diagnostics that reveal redundant assignment, allocation quality, and task-completion efficiency beyond aggregate return. STAT provides the testbed for this evaluation design.

4.1 Commitment-Constrained Spatial Task Allocation

Agents distribute themselves across spatially distributed tasks, commit to selected assignments, and complete all tasks efficiently. This induces a structured combinatorial coordination problem, where each assignment decision interacts with the choices of other agents, while commitment makes the effective action space state-dependent. As tasks are selected and completed, the set of meaningful assignment choices shrinks. Thus, the challenge is not only the nominal joint-action size, but whether agents make effective decisions at sparse, high-impact assignment points.

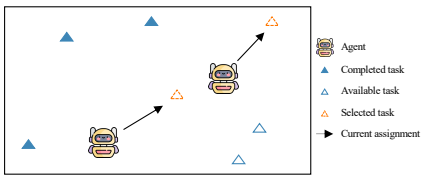


Figure 2: Illustration of STAT. Agents start at a fixed origin and must coordinate to complete spatially distributed tasks efficiently.

We use STAT, the Spatial Task Allocation Testbed, as a controlled environment for studying this setting (Figure 2). STAT is not intended to reproduce the full complexity of any single application domain. Instead, it isolates task-assignment coordination under controlled combinatorial scaling by allowing the number of agents, number of tasks, and spatial extent to be varied while holding task rules and observation access fixed. This makes STAT a testbed for examining whether return reflects coordination quality or if process-level diagnostics are needed to interpret performance. We further describe STAT in Section 5.1

4.2 Process-Level Diagnostics

Coordination-aware evaluation requires metrics that characterize *how* agents produce a given return. We therefore report task-performance metrics together with process-level diagnostics tailored to STAT's assignment structure. These diagnostics capture three important aspects of coordination in this setting: redundant assignment, allocation quality, and task-completion efficiency. Figure 3 illustrates the relationship between assignment conflicts and assignment diversity. Our task-performance metric is mean return, which captures the cumulative reward achieved by a method. To characterize coordination beyond return, we report total task assignment conflicts, conflict rate, conflicts per task, assignment diversity, and task completion throughput. We report all metrics over five random seeds using the mean and 95% confidence interval, following the evaluation protocol in Section 5.3.

Total task assignment conflicts. To measure redundant assignment, we count how many distinct tasks are selected by more than one agent before conflict resolution. Let S_t denote the multiset of task indices selected by agents at timestep t, where only task-selection actions are included. For each task j, let $n_t(j) = \sum_{s \in S_t} 1[s = j]$ denote the number of agents that selected task j. The timestep-level task assignment conflict count is $K_t = \sum_j 1[n_t(j) > 1]$, and the episode-level total conflict count is $K = \sum_{t=1}^{H} K_t$, where H is the episode horizon (length). This metric captures the breadth of redundant assignment. It counts the number of task identities experiencing conflict, but does not consider the number of agents involved in each conflict.

Conflict rate. Because longer episodes create more opportunities for conflict, we also report a timestep-normalized conflict rate, $K_{\text{rate}} = \frac{1}{H} \sum_{t=1}^{H} K_t$. This measures the average number of task assignment conflicts per timestep and helps distinguish methods that accumulate more conflicts simply because episodes last longer from methods that generate conflicts more frequently.

Conflicts per task. To compare settings with different task counts, we normalize conflicts by the number of tasks: This metric measures the density of conflict relative to the task set size, making conflict behavior more comparable across task-scaling experiments.

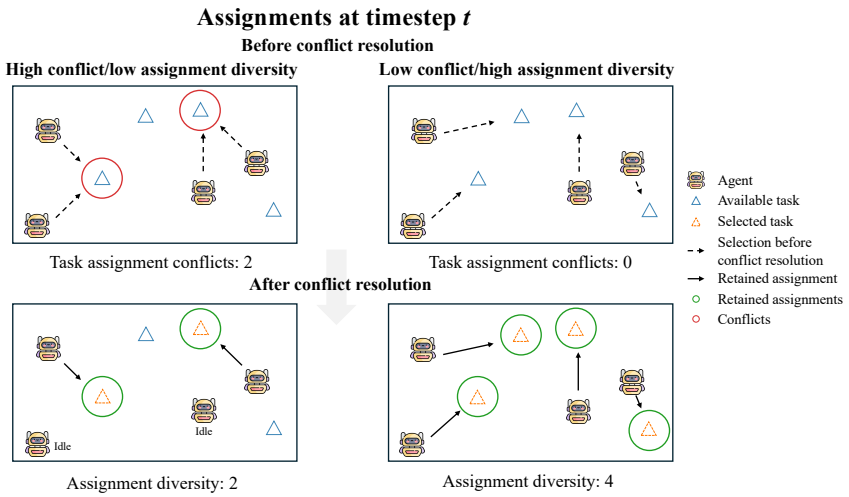


Figure 3: Illustration of the assignment-based process-level diagnostics used in this work. The top row shows task selections at timestep t before conflict resolution, and the bottom row shows the retained assignments after conflict resolution. **Task assignment conflicts** count the number of tasks selected by more than one agent before conflict resolution. **Assignment diversity** counts the number of distinct task assignments retained after conflict resolution.

Assignment diversity. To complement conflict metrics, we measure how broadly the team generates distinct new task assignments after conflict resolution. Let A_t denote the set of final agent actions at timestep t after conflict resolution. Since task-selection actions are indexed as $3+j$, corresponding to selecting task j, we define timestep-level assignment diversity as $D_t = \left|\{\, a-3 : a \in A_t,\ a \geq 3 \,\}\right|$. The episode-level assignment diversity is $\bar{D} = \frac{1}{H}\sum_{t=1}^{H} D_t$. This metric counts distinct newly retained task assignments at the current timestep. It does not measure all tasks currently being pursued, since agents may already be moving toward or executing tasks selected earlier. Higher values indicate the team more often produces diverse, non-redundant assignments at decision points.

Task completion throughput. We measure task-completion efficiency as the number of completed tasks per timestep: $\rho = \frac{M_{\text{completed}}}{H}$, where $M_{\text{completed}}$ is the number of tasks completed by the end of the episode. Throughput is not a pure conflict metric; rather, it helps distinguish poor assignment coordination from slow completion due to spatial scale, travel time, or long commitment phases.

These diagnostics are related but not redundant. Total conflicts, conflict rate, and conflicts per task capture redundant assignment before conflict resolution. Assignment diversity captures how broadly the team produces distinct retained assignments after conflict resolution. Throughput captures whether assignment decisions translate into completed tasks efficiently. Reporting these metrics together helps distinguish whether a method fails because agents select the same tasks, fail to distribute work broadly, or complete tasks slowly despite avoiding conflicts.

5 Experimental Setup And Analysis

5.1 STAT Environment and Commitment Structure

We use STAT to instantiate coordination-aware evaluation in commitment-constrained spatial task allocation. Building on the victim-tagging environment introduced in prior work [6, 7], STAT abstracts the core agent-task assignment structure into a domain-general testbed where agents, tasks, and environment size can be systematically varied while task rules and observation access remain fixed. In STAT, all agents begin at a fixed origin and tasks are distributed across a 2D grid. The environment is fully observable. The global state includes agent–task distance features, each agent's current mode, and task status variables indicating whether each task is available, assigned, or completed. We use full observability to avoid conflating coordination failures with unequal information access, so differences between methods primarily reflect how learning and action selection are structured across agents.

Table 1: STAT configurations used to evaluate controlled scaling behavior. The first three columns define the controlled scaling axes, and the remaining columns report derived quantities that affect assignment complexity and spatial density.

Problem Scale	# Agents	# Tasks	Environment Size	Timesteps for Training	Task Density (#T / env. area)	# Tasks per Agent (#T / #A)	Task Choices/Agent (#T)	# Joint Actions $\lvert\mathcal{A}\rvert = (\#T)^{\#A}$
Baseline	3	6	5×3	2M	0.400	2.0	6	216
	3	6	10×6	2M	0.100	2.0	6	216
	3	12	10×6	2M	0.200	4.0	12	1,728
	5	12	10×6	2M	0.200	2.4	12	248,832
Extreme	5	25	25×15	20M	0.067	5.0	25	9,765,625
	5	25	50×30	20M	0.017	5.0	25	9,765,625
	5	50	50×30	20M	0.033	10.0	50	312,500,000
	5	100	50×30	20M	0.067	20.0	100	10,000,000,000
	9	25	50×30	20M	0.017	2.78	25	3,814,697,265,625

Each agent has a discrete action space consisting of *idle*, *move*, *execute task*, and *select task*. The *select task* action expands into one action for each currently selectable task. Agents are governed by a finite-state commitment structure (Figure 5 in Appendix B). After selecting a task, an agent becomes committed to that assignment, moves toward the task until it is reached, executes the task for a fixed number of timesteps, and then returns to *select task* mode if selectable tasks remain, or *idle* otherwise. Thus, assignment decisions occur only at sparse decision points when agents are in *select task* mode, making each assignment choice high-impact.

Action masking enforces this commitment structure. Invalid actions are removed according to the agent's current mode and task status. For example, an agent that has not reached its assigned task cannot execute it, and an agent that is moving toward or executing a task cannot select a different task until its current commitment is resolved. Completed or already assigned tasks are also removed from the selectable task set. These masks remove invalid low-level choices so that the benchmark focuses on the high-level coordination problem of distributing agents across tasks.

When multiple agents select the same task at the same assignment timestep, STAT applies retrospective conflict resolution where the closest agent retains the assignment, while the others are forced to *idle*. The selected task is then treated as assigned and is no longer selectable. This makes redundant allocation observable as a process-level coordination failure. Conflict wastes assignment opportunities and delays task completion, rather than only appearing indirectly through lower return. Under this design, the assignment-level joint action space scales as $m_t^{n_t}$, where m_t is the number of currently selectable tasks and n_t is the number of agents currently in *select task* mode. The effective action space is therefore state-dependent. As agents commit to tasks and as tasks become assigned or completed, fewer assignment choices remain. Additional details and formulation are provided in Appendix B.

5.2 Methods Evaluated

We evaluate representative value-based cooperative MARL methods as probes for coordination-aware evaluation under scale. The methods span different assumptions about coordination. Centralized Training with Centralized Execution (CTCE) methods can reason over joint decisions but scale poorly, Decentralized Training with Decentralized Execution (DTDE) methods are scalable but do not explicitly model inter-agent dependencies, and Centralized Training with Decentralized Execution (CTDE) methods seek a middle ground. We include CTCE, CTDE, and DTDE approaches to test how different training and execution structures affect process-level coordination diagnostics. These include DQN [22], FDQN [7], VDN [31], QMIX [27], QTRAN [29], and IQL [32]. Additional method details are in Appendix C.

5.3 Scaling Configurations and Evaluation Protocol

STAT supports controlled scaling along three axes: number of agents, number of tasks, and environment size. Increasing the number of agents increases the number of simultaneous assignment decisions, which can improve parallel task completion but also raises the risk of redundant allocation. Increasing the number of tasks expands the assignment choice set and increases the number of possible agent–task allocations. Increasing environment size changes the distance structure of the problem, affecting travel time and task-completion efficiency.

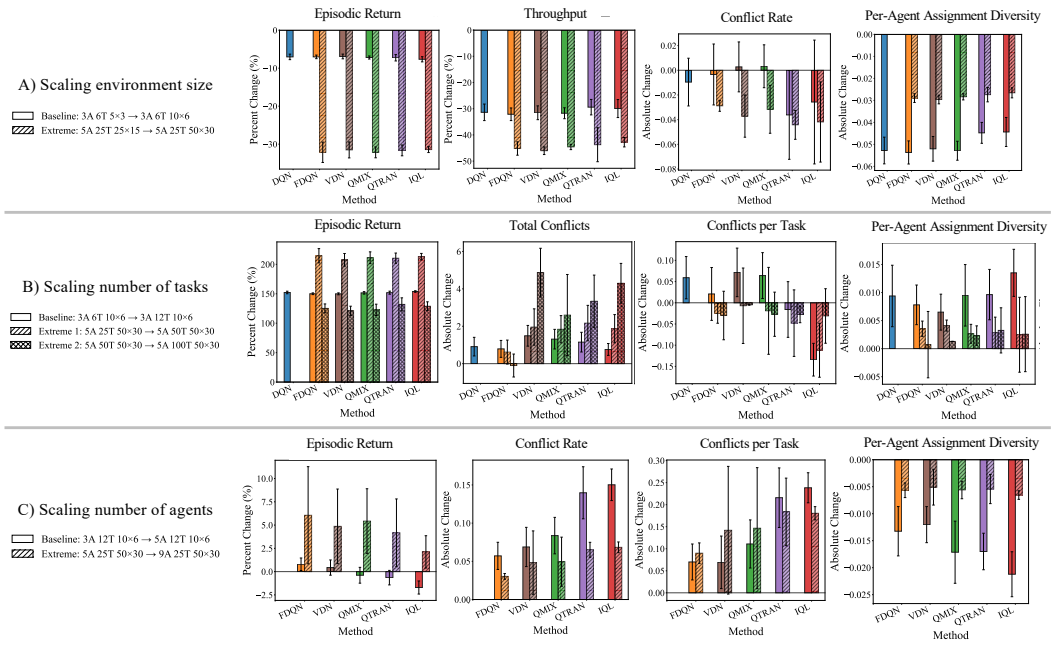


Figure 4: Coordination-aware scaling analysis. Each row isolates one scaling axis: (A) environment size, (B) number of tasks, and (C) number of agents. Bars show mean changes across five seeds with 95% confidence intervals. Return alone gives an incomplete picture of scaling behavior.

We construct benchmark configurations by varying one axis at a time, allowing us to compare how outcome-level performance and process-level diagnostics change under different forms of scale. These controlled comparisons help distinguish whether performance changes are driven by spatial efficiency, assignment pressure, or increased simultaneous decision-making. Table 1 summarizes the nine STAT configurations used in our benchmark. The Baseline regime contains smaller settings where most methods are expected to learn reasonable task-completion behavior, while the Extreme regime creates substantially larger assignment spaces and stronger coordination demands. Standard DQN is included only in the three smallest configurations because its fully centralized output layer enumerates the joint action space and becomes computationally infeasible as the number of agents and task choices increases. We therefore include DQN where tractable as a reference for unconstrained centralized joint-action reasoning, but omit it from larger-scale comparisons.

To ensure a fair comparison across methods, we use the same training budget and evaluation protocol within each regime. We train each method for 2 million environment timesteps on the Baseline settings and 20 million environment timesteps on the Extreme settings. Each run uses one A100 GPU. During training, we evaluate the current policy every 10,000 environment timesteps using 20 test episodes. We run each method/configuration with five training seeds.

For each evaluation checkpoint, we record the task-performance, process-level coordination, and computational-efficiency metrics defined in Section 4.2. For metric X, we report the average at checkpoint t for experiment j as $\bar{X}_{t,j} = \frac{1}{P}\sum_{i=1}^{P} X_{t,i,j}$, where $P = 20$ is the number of test episodes and $X_{t,i,j}$ denotes the value of metric X in the i-th test episode at checkpoint t for experiment j. Unless otherwise stated, reported curves and tables summarize performance across five seeds using the mean and 95% confidence interval. Additional implementation and hyperparameter-tuning details are provided in Appendix D.

5.4 Systematic Scaling Benchmark Analysis

We analyze coordination under three controlled scaling interventions: environment size, number of tasks, and number of agents. Using matched configurations from the Baseline and Extreme regimes in Table 1, we vary one axis at a time to separate changes driven by spatial efficiency, assignment pressure, and simultaneous decision-making. Figure 4 summarizes each scaling intervention using

the core metrics defined in Section 4.2. Table 14 in Appendix G reports statistical significance and direction for the changes across each setting. Additional supporting mechanism-level diagnostics, computational efficiency, and exploratory COMA results and are reported in Appendix H, Appendix J, and Appendix K respectively.

Scaling Environment Size: Increasing environment size primarily changes the spatial structure of the problem. Agents must travel farther before completing tasks, so changes in return may reflect task-completion efficiency, assignment coordination, or both. Figure 4A separates these effects. When grid size increases, return and throughput decrease substantially. Per-agent assignment diversity also decreases, indicating that agents generate fewer distinct new assignments per unit time. This is expected because agents spend longer periods committed to movement or execution, reducing how often they return to sparse assignment decision points. These trends show why return alone is insufficient under spatial scaling. The decrease in return does not necessarily imply more redundant assignment; the process-level diagnostics show that performance loss is largely associated with lower throughput and fewer assignment opportunities. Meanwhile, conflict-rate changes are smaller and may even decrease because agents have fewer opportunities to conflict. Together, throughput, conflict rate, and per-agent assignment diversity distinguish spatial inefficiency from assignment-level coordination failure.

Scaling Number of Tasks: Increasing the number of tasks expands each agent's assignment choice set and increases the number of possible agent–task allocations. This directly stresses combinatorial assignment pressure. Figure 4B shows that return generally increases as task count grows, since additional tasks create more opportunities for reward. However, this outcome-level improvement masks a simultaneous degradation in the coordination process: total conflicts also increase, indicating that agents more often select overlapping tasks as the assignment space expands. The contrast is most visible in the second extreme comparison, where task count increases from 50 to 100. Conflicts continue to rise, but the return gains are smaller than in earlier task-scaling comparisons. This suggests that adding tasks initially improves productivity by increasing the number of available task-completion opportunities, but at larger scales the added coordination burden begins to offset these gains. Thus, higher task count can make the benchmark look easier from return alone while making the underlying assignment problem more coordination-limited. The normalized diagnostics further clarify this behavior. Conflicts per task indicate whether contention grows relative to the size of the task set, rather than only in absolute terms. In Figure 4B, conflicts per task are mixed and sometimes decrease, suggesting that some conflict growth is absorbed by the larger task set. However, this does not mean coordination improves since total conflicts still rise, so agents accumulate more redundant assignments overall. Per-agent assignment diversity changes only modestly, indicating that agents do not consistently use the expanded task set to produce substantially broader division of labor. These metrics show that higher return can coexist with increasing redundant assignment, and at extreme task counts these coordination costs begin to limit further performance gains.

Scaling Number of Agents: Increasing the number of agents increases potential parallelism, but also produces the largest combinatorial growth in the nominal joint action space. Agent scaling therefore tests whether methods can convert additional team capacity into coordinated work, or whether added decision-makers amplify redundant assignment. Figure 4C shows that adding agents does not uniformly improve performance. In the Baseline comparison, return decreases for several methods, indicating that additional agents can hurt performance when the task load is not large enough to offset the added coordination burden. In the Extreme comparison, return improves more consistently because there are enough tasks for added agents to provide useful parallelism. However, these gains are method-dependent, showing that additional agents help only when a method can translate increased team capacity into coordinated assignments. The process-level diagnostics explain this pattern. Conflict rate and conflicts per task increase under agent scaling, especially in the Baseline comparison, showing that additional agents create more overlapping selections among the same task set. Per-agent assignment diversity also decreases or changes only modestly, indicating that the added agents do not necessarily produce proportionally broader division of labor. Thus, higher agent count can increase parallel capacity while simultaneously reducing coordination efficiency.

These results show why return alone is insufficient for evaluating agent scaling. A higher-return policy may still waste assignment opportunities through conflicts, while a lower-return policy may fail because added agents amplify redundant decisions rather than useful parallelism. Conflict rate, conflicts per task, and per-agent assignment diversity reveal whether additional agents improve coordinated parallelism or simply increase simultaneous decision pressure.

6 Discussion and Conclusion

We present a coordination-aware evaluation perspective for cooperative MARL under combinatorial scaling. Rather than evaluating methods only by aggregate return, we argue that benchmarks should also report process-level diagnostics that reveal how agents coordinate. We instantiate this perspective using STAT, a controlled commitment-constrained spatial task-allocation testbed that supports systematic scaling over agents, tasks, and environment size while making assignment conflicts, assignment diversity, and task-completion throughput directly observable.

The benchmark analysis supports three conclusions. First, return alone is insufficient for diagnosing cooperative behavior under scale. Similar returns can correspond to different coordination efficiency, measured in this benchmark through conflict rates, conflicts per task, assignment diversity, and throughput. Second, performance becomes increasingly coordination-limited as problem complexity grows. Environment-size scaling primarily reduces throughput and assignment opportunities, task scaling increases absolute assignment conflict while eventually limiting return gains, and agent scaling increases simultaneous decision pressure. These results show that scaling is governed not only by nominal action-space size, but also by how methods handle assignment pressure, sparse commitment decisions, and inter-agent dependence. Third, method structure shapes these failures. Centralized or factorized reasoning can reduce redundant assignment when tractable, CTDE methods remain scalable and competitive across settings, and independent learning is most vulnerable to redundant assignment in highly interdependent settings (see more in Appendix I). These findings support coordination-aware evaluation for cooperative MARL and indicate that benchmarks should evaluate not only what return agents achieve, but how agents coordinate to achieve it.

Limitations and Future Work. This work studies coordination-aware evaluation in a controlled commitment-constrained spatial task-allocation setting. STAT intentionally holds observation access and task rules fixed, abstracts away low-level collision avoidance and path planning, and isolates assignment coordination as the primary coordination bottleneck. These choices make process-level failures directly measurable, but they also limit the scope of the conclusions. STAT does not capture partial observability, explicit communication, heterogeneous agent capabilities, stochastic task arrivals, congestion, or richer movement dynamics. Future work could extend the same evaluation protocol to controlled variants with these factors while preserving interpretable process diagnostics. Another direction is to relax STAT's action-masking and finite-state commitment structure, allowing agents to learn when to replan, abandon commitments, or recover from inefficient choices.

Our empirical evaluation focuses primarily on value-based MARL methods spanning CTCE, CTDE, and DTDE paradigms. These methods serve as useful probes for studying how training and execution structure affect coordination under scale, but they do not cover the full space of cooperative MARL algorithms. Extending the evaluation to actor-critic, communication-based, transformer-based, planning-learning hybrid, and combinatorial-action methods would provide a broader view of how different algorithmic families handle assignment pressure and sparse commitment decisions. Exploratory COMA results are included in the appendix, but a more complete evaluation of policy-gradient and on-policy methods remains important future work. Finally, applying the same process-level evaluation lens to other cooperative MARL benchmarks would help determine which coordination diagnostics generalize across domains and which are specific to commitment-constrained spatial task allocation.

Overall, this work highlights the importance of evaluating how cooperative agents coordinate, not only what return they achieve. Coordination-aware diagnostics provide a more interpretable view of failure modes under scale and can help future benchmarks distinguish task performance from the coordination processes that produce it.

Acknowledgments and Disclosure of Funding

This work was supported by the National Science Foundation Graduate Research Fellowship Program under grant number 2234693. We used an icon from FlatIcon by author Freepik.

References

[1] Aakriti Agrawal, Amrit Singh Bedi, and Dinesh Manocha. Rtaw: An attention inspired reinforcement learning method for multi-robot task allocation in warehouse environments. *arXiv preprint arXiv:2209.05738*, 2022.

[2] Aakriti Agrawal, Senthil Hariharan, Amrit Singh Bedi, and Dinesh Manocha. Dc-mrta: Decentralized multi-robot task allocation and navigation in complex environments. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022.

[3] Sofia Amador, Steven Okamoto, and Roie Zivan. Dynamic multi-agent task allocation with spatial and temporal constraints. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, pages 1384–1390, 2014.

[4] Upasana Biswas, Vardhan Palod, Siddhant Bhambri, and Subbarao Kambhampati. Who is helping whom? analyzing inter-dependencies to evaluate cooperation in human-ai teaming. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(21):17347–17356, 2026.

[5] Lorenzo Canese, Gian Carlo Cardarilli, Luca Di Nunzio, Rocco Fazzolari, Daniele Giardino, Marco Re, and Sergio Spanò. Multi-agent reinforcement learning: A review of challenges and applications. *Applied Sciences*, 11(11):4948, 2021.

[6] Maria Ana Cardei and Afsaneh Doryab. Practical heuristics for victim tagging during a mass casualty incident emergency medical response. In *2024 IEEE 20th International Conference on Automation Science and Engineering (CASE)*, pages 165–172, 2024.

[7] Maria Ana Cardei and Afsaneh Doryab. Factorized deep q-network for cooperative multi-agent reinforcement learning in victim tagging. *IEEE Transactions on Automation Science and Engineering*, 23:3109–3120, 2026.

[8] Micah Carroll, Rohin Shah, Mark K. Ho, Thomas L. Griffiths, Sanjit A. Seshia, Pieter Abbeel, and Anca Dragan. *On the utility of learning about humans for human-AI coordination*. Curran Associates Inc., Red Hook, NY, USA, 2019.

[9] Filippos Christianos, Lukas Schäfer, and Stefano V Albrecht. Shared experience actor-critic for multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[10] Daniel Claes, Philipp Robbel, Frans A. Oliehoek, Karl Tuyls, Daniel Hennes, and Wiebe van der Hoek. Effective approximations for multi-robot coordination in spatially distributed tasks. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, pages 881–890, Richland, SC, 2015. International Foundation for Autonomous Agents and Multiagent Systems.

[11] Zhenhui Feng, Renbin Xiao, and Mingzhi Xiao. Spatial crowdsourcing task allocation for heterogeneous multi-task hybrid scenarios: A model-embedded role division approach. *Frontiers of Information Technology & Electronic Engineering*, 26:1144–1163, 2025.

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

[13] Brian P Gerkey and Maja J Matarić. A formal analysis and taxonomy of task allocation in multi-robot systems. *The International journal of robotics research*, 23(9):939–954, 2004.

[14] Pablo Hernandez-Leal, Bilal Kartal, and Matthew E Taylor. A survey and critique of multiagent deep reinforcement learning. *Autonomous Agents and Multi-Agent Systems*, 33(6):750–797, 2019.

[15] Siyi Hu, Fengda Zhu, Xiaojun Chang, and Xiaodan Liang. Policy diagnosis via measuring role diversity in cooperative multi-agent reinforcement learning. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 9041–9071. PMLR, 2022.

[16] Aleksandar Krnjaic, Raul D. Steleac, Jonathan D. Thomas, Georgios Papoudakis, Lukas Schäfer, Andrew Wing Keung To, Kuan-Ho Lao, Murat Cubuktepe, Matthew Haley, Peter Börsting, and Stefano V. Albrecht. Scalable multi-agent reinforcement learning for warehouse logistics with robotic and human co-workers. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 677–684, 2024.

[17] Kun Li, Shengling Wang, Hongwei Shi, Xiuzhen Cheng, and Minghui Xu. Spatial crowdsourcing task allocation scheme for massive data with spatial heterogeneity. *arXiv preprint arXiv:2310.12433*, 2023.

[18] Michael L. Littman. Markov games as a framework for multi-agent reinforcement learning. In *Proceedings of the Eleventh International Conference on Machine Learning*, pages 157–163, 1994.

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

[20] Anuj Mahajan, Tabish Rashid, Mikayel Samvelyan, and Shimon Whiteson. Maven: Multi-agent variational exploration. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[21] Rajiv T. Maheswaran, Pedro A. Szekely, Marcel Becker, Stephen Fitzpatrick, Gergely Gati, Jing Jin, Robert Neches, Narges Noori, Craig Milo Rogers, Romeo Sanchez, Kevin Smyth, and Chris VanBuskirk. Predictability and criticality metrics for coordination in complex environments. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems*, volume 2, pages 647–654, 2008.

[22] 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, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

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

[24] Afshin Oroojlooy and Davood Hajinezhad. A review of cooperative multi-agent deep reinforcement learning. *Applied Intelligence*, 53:13677–13722, 2023.

[25] George Papadopoulos, Andreas Kontogiannis, Foteini Papadopoulou, Chaido Poulianou, Ioannis Koumentis, and George Vouros. An extended benchmarking of multi-agent reinforcement learning algorithms in complex fully cooperative tasks. In *Proceedings of the 24th International Conference on Autonomous Agents and Multiagent Systems*, AAMAS '25, page 1613–1622, Richland, SC, 2025. International Foundation for Autonomous Agents and Multiagent Systems.

[26] Georgios Papoudakis, Filippos Christianos, Lukas Schäfer, and Stefano V. Albrecht. Benchmarking multi-agent deep reinforcement learning algorithms in cooperative tasks. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks (NeurIPS)*, 2021.

[27] Tabish Rashid, Mikayel Samvelyan, Christopher de Witt, Gregory Farquhar, Jakob N 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 (ICML)*, volume 80, pages 4295–4304. PMLR, 2018.

[28] Mikayel Samvelyan, Tabish Rashid, Christian Schroeder de Witt, Gregory Farquhar, Nantas Nardelli, Tim G J Rudner, Philip H S Torr, Jakob Foerster, and Shimon Whiteson. The starcraft multi-agent challenge. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, pages 2186–2188, 2019.

[29] Kyunghwan Son, Daewoo Kim, Wan Ju Kang, David Hostallero, and Yung Yi. Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, pages 5887–5896. PMLR, 2019.

[30] Younes Strittmatter, Rachael Skye, Samuel Lozano Iglesias, Samuel Liebana, Andrew Saxe, Miguel Ruiz-Garcia, Erin Teich, and Markus Spitzer. When collaboration beats ability: Mixed-ability teams can outperform high-ability teams under coordination demands. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, 2026.

[31] Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech M. Czarnecki, Vinícius Flores Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z. Leibo, Karl Tuyls, and Thore Graepel. Value-decomposition networks for cooperative multi-agent learning. *ArXiv*, abs/1706.05296, 2017.

[32] Ming Tan. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proceedings of the Tenth International Conference on Machine Learning (ICML 1993)*, pages 330–337, San Francisco, CA, USA, 1993. Morgan Kaufmann.

[33] Jianhao Wang, Zhizhou Ren, Terry Liu, Yang Yu, and Chongjie Zhang. QPLEX: Duplex dueling multi-agent q-learning. In *International Conference on Learning Representations (ICLR)*, 2021.

[34] B. L. Welch. The generalization of Student's problem when several different population variances are involved. *Biometrika*, 34(1/2):28–35, 1947.

[35] Guanyu Ye, Yan Zhao, Xuanhao Chen, and Kai Zheng. Task allocation with geographic partition in spatial crowdsourcing. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 2404–2413, 2021.

[36] Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. The surprising effectiveness of ppo in cooperative multi-agent games. In *Advances in Neural Information Processing Systems*, volume 35, pages 24611–24624, 2022.

[37] Yongchao Zhang, Qingyu Yang, Dou An, and Weidong Chen. Coordination between individual agents in multi-agent reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 11387–11394, 2021.

[38] Yan Zhao, Xuanlei Chen, Guanyu Ye, Fangda Guo, Kai Zheng, and Xiaofang Zhou. Task allocation in spatial crowdsourcing: An efficient geographic partition framework. *IEEE Transactions on Knowledge and Data Engineering*, 36(9):4943–4955, 2024.

[39] Yiheng Zhu, Yang Zhan, Xuankun Huang, Yuwei Chen, Jiangwen Wei, Wei Feng, Yinzhi Zhou, Haoyuan Hu, Jieping Ye, et al. Ofcourse: A multi-agent reinforcement learning environment for order fulfillment. *Advances in Neural Information Processing Systems*, 36:34765–34777, 2023.

A Code Release

We release the STAT environment together with executable training and evaluation code for all methods considered in this paper. The release includes DQN and FDQN training/evaluation scripts, PyMARL integration for IQL, VDN, QMIX, QTRAN, and COMA, smoke-test configurations, and example commands for running methods on configurable STAT instances. This artifact is intended to support reproducibility, executable verification, and future extensions of STAT with additional methods or environment variants. Code is available at `https://github.com/mariacardei/coordination_aware_MARL`.

B STAT Environment Details

We provide additional details on STAT, the Spatial Task Allocation Testbed, used to instantiate coordination-aware evaluation in commitment-constrained spatial task allocation. STAT generalizes the victim-tagging environment used in prior work [7, 6] into a domain-general spatial task-allocation testbed. Unlike the prior application-specific formulation, we use STAT to systematically vary agents, tasks, and spatial extent, and to evaluate process-level coordination diagnostics under controlled combinatorial scaling. STAT provides a controlled setting in which task rules and observation access remain fixed while coordination pressure changes with scale.

B.1 Environment Parameters, State, and Observability

A STAT instance is specified by

$$\mathcal{E} = \langle n, m, W, H, K, v, \Theta_R \rangle,$$

where n is the number of agents, m is the number of tasks, $W \times H$ defines the spatial grid, K is the number of timesteps required to execute a task after arrival, v is the agent movement speed, and $\Theta_R = \{R_0, \eta, \beta, \alpha, \lambda_{\text{step}}\}$ denotes the reward parameters. In the current benchmark, agents and tasks are homogeneous, but the same formulation can be extended to agent-specific speeds, task-specific execution times, heterogeneous task requirements, or partial observability.

Let $\mathcal{N} = \{1, \ldots, n\}$ denote the set of agents and $\mathcal{M} = \{1, \ldots, m\}$ denote the set of tasks. All agents begin from a fixed origin, and tasks are spatially distributed across a 2D grid. At timestep t, agent $i \in \mathcal{N}$ has position $x_i(t) \in [0, W] \times [0, H]$, mode $q_i(t)$, and current assignment $g_i(t) \in \mathcal{M} \cup \{\varnothing\}$. Each task $j \in \mathcal{M}$ has a fixed location y_j and status

$$z_j(t) \in \{\text{AVAILABLE}, \text{ASSIGNED}, \text{COMPLETED}\}.$$

The team objective is to complete all tasks efficiently. An episode terminates when all tasks are completed or when the environment reaches the maximum episode length.

We use a fully observable setting so that comparisons across training and execution paradigms are not confounded by differences in information access. The global state includes agent–task distance information, each agent's current mode, and task status variables indicating whether each task is available, assigned, or completed. Although STAT is fully observable in this benchmark, its structure naturally supports partially observable variants in future work.

B.2 Action Space and Finite-State Commitment

Each agent has a discrete action space consisting of IDLE, MOVE, EXECUTE, and SELECT actions. The SELECT action expands into one action for each task:

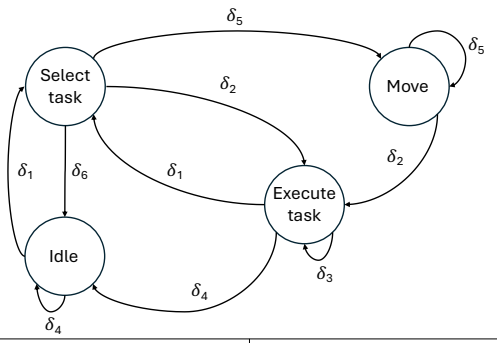
$$\mathcal{A}_i = \{\text{IDLE}, \text{MOVE}, \text{EXECUTE}\} \cup \{\text{SELECT}(j) : j \in \mathcal{M}\}.$$

Thus, the nominal action set has size $3 + m$. The valid action set is state-dependent and is enforced through action masking, as described in the next subsection.

Agents are governed by a finite-state commitment structure (Figure 5). We represent each agent's mode as

$$q_i(t) \in \{\text{IDLE}, \text{SELECT TASK}, \text{MOVE}, \text{EXECUTE TASK}\}.$$

The SELECT TASK mode is the decision mode in which an agent may choose among currently selectable tasks. Once a task is selected and retained after conflict resolution, the agent becomes



δ_1: incomplete selectable tasks remain	δ_4: no selectable tasks remain
δ_2: assigned task reached	δ_5: assigned task not yet reached; move one step toward task
δ_3: task execution in progress; decrement remaining time	δ_6: lost conflict resolution

Figure 5: Finite-state commitment structure representing agent modes and valid transitions.

committed to that task and transitions to MOVE. The agent remains in MOVE while it advances toward the assigned task. Once the assigned task is reached, the agent transitions to EXECUTE TASK, where it remains for a fixed number of timesteps K until the task is completed. After execution, the agent returns to SELECT TASK if incomplete selectable tasks remain, or transitions to IDLE if no selectable tasks remain. If an agent loses conflict resolution after selecting a task, it transitions to IDLE for that timestep.

This design makes STAT a commitment-constrained task-allocation problem. The nominal assignment space is combinatorial, but the effective action space is state-dependent. As agents commit to tasks and as tasks become assigned or completed, the set of selectable tasks decreases. Therefore, the complexity of the assignment problem changes over the episode as early timesteps may contain many feasible task assignments, while later timesteps contain fewer meaningful choices. This property concentrates coordination pressure at sparse, high-impact decision points.

B.3 Action Masking and State-Dependent Assignment Complexity

STAT uses action masks to enforce the finite-state commitment structure and remove invalid actions. Let $\mathcal{V}_i(s_t) \subseteq \mathcal{A}_i$ denote the valid action set for agent i in state s_t. If agent i is in SELECT TASK mode, then its valid task-selection actions are $\mathcal{V}_i(s_t) = \{\text{SELECT}(j) : z_j(t) = \text{AVAILABLE}\}$. If agent i is in MOVE mode and has not yet reached its assigned task, then the valid action set is restricted to $\mathcal{V}_i(s_t) = \{\text{MOVE}\}$. If agent i is in EXECUTE TASK mode, then the valid action set is restricted to $\mathcal{V}_i(s_t) = \{\text{EXECUTE}\}$. If no selectable tasks remain, the valid action set is restricted to $\mathcal{V}_i(s_t) = \{\text{IDLE}\}$. Completed or already assigned tasks are removed from the set of selectable task actions.

The masking removes invalid low-level choices so that the benchmark focuses on the high-level coordination problem of distributing agents across tasks. Without masking, a substantial part of the learning problem would involve discovering which actions are invalid in each state. With masking, the core challenge becomes whether agents make compatible assignment decisions when meaningful choices are available.

At an assignment decision point, let m_t be the number of selectable tasks and n_t be the number of agents currently in SELECT TASK mode. The effective assignment-level joint action space is

$$|\mathcal{A}_{\text{assign}}(t)| = m_t^{n_t}.$$

Thus, the assignment space grows combinatorially with the number of agents simultaneously making assignment decisions and the number of selectable tasks. Unlike settings with a fixed joint action space, both m_t and n_t change throughout an episode. Tasks become assigned or completed, and agents become temporarily committed to movement or execution. STAT therefore induces a state-dependent combinatorial action space whose complexity generally decreases as tasks are assigned and completed.

B.4 Conflict Resolution

Multiple agents may select the same task at the same assignment timestep. STAT resolves these assignment conflicts retrospectively. Let

$$S_j(t) = \{i \in \mathcal{N} : a_i(t) = \textsc{select}(j)\}$$

be the set of agents that select task j at timestep t. If $|S_j(t)| > 1$, the retained agent is

$$i^*(j,t) = \arg\min_{i \in S_j(t)} d(x_i(t), y_j),$$

with ties broken deterministically by agent index. Agent $i^*(j,t)$ receives assignment $g_{i^*}(t) = j$, while all other agents in $S_j(t) \setminus \{i^*(j,t)\}$ default to IDLE. The selected task is then marked as ASSIGNED and is no longer available for future selection.

This conflict-resolution rule makes coordination failures explicit and measurable. In STAT, a conflict is not only reflected indirectly through lower reward, it is an observable process-level event that reveals redundant allocation. Agents that lose conflict resolution do not make progress during that timestep, making redundant assignment costly through lost opportunity and delayed task completion. This allows the benchmark to distinguish policies that achieve similar return but differ in how efficiently they distribute agents across tasks.

B.5 Movement, Execution, and Abstractions

After an agent receives a task assignment, it moves toward the selected task. For an agent in MOVE mode, the position update is

$$x_i(t+1) = x_i(t) + \min\{v, d(x_i(t), y_{g_i(t)})\} \frac{y_{g_i(t)} - x_i(t)}{d(x_i(t), y_{g_i(t)})},$$

where $d(\cdot,\cdot)$ denotes Euclidean distance. When the agent reaches the assigned task location, it enters EXECUTING. After K execution timesteps, the task is marked COMPLETED, and the agent becomes available to select another task if any remain.

STAT abstracts away explicit agent–agent collision dynamics and low-level path-planning constraints. Agents move toward their assigned tasks without needing to solve a separate pathfinding problem. This abstraction prevents collision avoidance or complex navigation from becoming the dominant source of difficulty. The benchmark instead isolates spatial task-assignment coordination under combinatorial scaling.

B.6 Reward Function

We use a fixed reward function across all STAT configurations to provide a consistent task objective as problem complexity scales. Each agent $i \in \mathcal{N}$ receives an individual reward $\mathcal{R}_i(t)$ at time t, and the total team reward is

$$\mathcal{R}_{\text{total}}(t) = \sum_{i \in \mathcal{N}} \mathcal{R}_i(t).$$

At each timestep, an agent receives a step penalty $\mathcal{R}_i(t) = -\lambda_{\text{step}}$, unless it completes a task. When an agent completes a task, it instead receives

$$\mathcal{R}_i(t) = \mathcal{R}^b(t)\left(1 + \alpha T_{\text{completed}}(t)\right),$$

where $T_{\text{completed}}(t)$ is the total number of tasks completed by time t, and α controls the progressive bonus for cumulative task completion. The base reward decays with elapsed time:

$$\mathcal{R}^b(t) = R_0 - \eta \left\lfloor \frac{\text{steps}_t}{\beta} \right\rfloor,$$

where R_0 is the initial base reward, η is the penalty applied at each decay interval, and β is the interval length in timesteps. In our experiments, we set $R_0 = 30$, $\eta = 0.5$, $\beta = 10$, $\alpha = 0.1$, and $\lambda_{\text{step}} = 1$, following prior work [7]. We keep these reward parameters fixed across all benchmark settings so that methods optimize the same task objective.

Table 2: Relevant benchmarks for cooperative MARL evaluation. ✓, ✓*, and – denote direct support, partial or configuration-dependent support, and not a primary focus, respectively.

Benchmark	Systematic Combinatorial Scaling	Built-in Process Metrics	Isolated Coord. Failure Mode	Sparse High-Impact Decisions	Main Coordination Bottleneck
SMAC [28]	✓*	–	–	–	Decentralized micromanagement under partial observability
MPE [19]	✓*	–	–	–	Particle-world coordination, communication, and competition
LBF [9]	✓	✓*	–	–	Cooperative foraging and capability matching
Overcooked [8]	✓*	✓*	–	–	Collaborative planning and division of labor
RWARE [26]	✓	✓*	–	–	Warehouse routing, pickup, and delivery coordination
STAT (ours)	✓	✓	✓	✓	Commitment-constrained spatial task assignment

B.7 Environment Scope

STAT has the broad structure of a grid-based cooperative task-allocation problem, making it relevant to domains such as warehouse logistics, order fulfillment, delivery, victim tagging, and search and rescue [25, 16, 7, 39]. The current benchmark intentionally uses homogeneous agents, homogeneous tasks, full observability, and simplified movement. This controlled scope is chosen to isolate structured combinatorial coordination without introducing additional confounds such as heterogeneous capabilities, partial observability, complex perception, collision avoidance, or domain-specific execution mechanics.

These simplifications are also natural extension points. STAT could be extended to heterogeneous agents by varying speed, specialization, or sensing capabilities; to heterogeneous tasks by varying execution time, priority, or completion requirements; and to partially observable settings by restricting each agent's access to the global state. In this work, we keep these factors fixed so that the benchmark specifically tests how coordination-aware metrics change as agents, tasks, and spatial scale are varied.

Table 2 compares STAT with commonly used cooperative MARL benchmarks along dimensions central to coordination-aware evaluation under scale. Existing benchmarks provide rich testbeds for cooperative behavior, while STAT is designed to complement them by isolating assignment coordination and exposing process-level coordination failures that may be hidden by return, success rate, win rate, or completion time alone.

C Methods Evaluated

Table 3: Comparison of algorithms across training schemes, including their descriptions, advantages, limitations, and roles in the benchmark.

Method	Training Scheme	Description	Advantages	Limitations	Role in Benchmark
DQN [22]	CTCE	Fully centralized Q-learning over the full joint action space.	No factorization assumptions; fully expressive.	Poor scalability due to exponential joint action growth.	Upper-bound baseline for full joint reasoning.
FDQN [7]	CTCE	Centralized Q-learning with a factorized action representation.	Handles large combinatorial spaces; captures dependencies.	Requires centralized execution; less scalable.	Evaluates centralized control with structured action decomposition.
VDN [31]	CTDE	Decomposes the joint Q-value as the sum of individual agent Q-values.	Scalable; enables some coordination via shared reward.	Cannot model agent interactions; assumes additivity / weak dependence.	Tests the limits of additive factorization in structured settings.
QMIX [27]	CTDE	Learns a joint Q-function via monotonic mixing of individual agent Q-values.	Captures limited dependencies; strong empirical performance.	Monotonic constraint restricts expressivity.	Evaluates coordination under constrained interaction modeling.
QTRAN [29]	CTDE	Learns an unconstrained joint Q-function with consistency constraints for decentralized execution.	More expressive; can model complex dependencies.	Hard to train; unstable; higher optimization complexity.	Tests whether greater expressivity improves performance in combinatorial settings.
IQL [32]	DTDE	Each agent learns an independent Q-function using local observations.	Simple, scalable, easy to implement.	Limited coordination; non-stationarity; ignores dependencies.	Baseline for fully decentralized learning without coordination.

We evaluate representative value-based cooperative MARL methods as probes for coordination-aware evaluation under scale. The methods span different assumptions about coordination. Centralized Training with Centralized Execution (CTCE) methods can reason over joint decisions but scale poorly, Decentralized Training with Decentralized Execution (DTDE) methods are scalable but do not explicitly model inter-agent dependencies, and Centralized Training with Decentralized Execution (CTDE) methods seek a middle ground [32, 31, 27, 29]. We include CTCE, CTDE, and DTDE approaches to test how different training and execution structures affect process-level coordination diagnostics. Table 3 summarizes the evaluated algorithms and their roles in the benchmark. Exploratory COMA [12] results are reported in Appendix C.1.

Centralized Training and Centralized Execution In the CTCE paradigm, both learning and action selection are performed centrally over the full multi-agent system. We employ a DQN and FDQN.

DQN. We extend Deep Q-Networks (DQN) [22], originally proposed for single-agent reinforcement learning, to a fully centralized multi-agent setting. Specifically, we formulate the full multi-agent system as a single centralized learner over the global state and joint action space. A centralized Q-network is then trained to estimate the value of each joint action for the full system state, enabling direct reasoning over joint decisions without factorization assumptions, but scaling poorly as the size of the joint action space grows exponentially with the number of agents.
FDQN. In Factorized Deep Q-Networks (FDQN) [7], the centralized joint action-value function is learned using a factorized representation of the joint action space. This preserves centralized training and execution while improving scalability relative to standard DQN, enabling more efficient learning in structured combinatorial settings. We categorize FDQN as CTCE because its factorization is used to represent a centralized joint value function rather than to enable decentralized action selection.

Centralized Training and Decentralized Execution In the CTDE paradigm, agents use centralized or joint information during training but select actions independently at execution time. We leverage VDN, QMIX, and QTRAN.
VDN. In Value Decomposition Networks (VDN) [31], the joint action-value function is decomposed as the sum of individual agent Q-functions. Each agent maintains its own utility network, and the joint action-value function is trained using a DQN-style temporal-difference loss [22], with gradients from the joint loss backpropagated to each agent's network.
QMIX. In QMIX [27], the joint action-value function is computed by a monotonic mixing network that combines the individual agent Q-functions into a global Q-function. This extends VDN to more complex settings while preserving decentralized execution, since the joint argmax is consistent with the individual argmax actions of each agent. The model is trained using a DQN-style loss, with gradients backpropagated through the mixing network to the individual agent utilities.
QTRAN. In QTRAN [29], the joint action-value function is learned using a more general factorization that relaxes the monotonicity constraint imposed by QMIX. It introduces consistency constraints to align a centralized joint Q-function with decentralized action selection, enabling a more expressive representation of inter-agent dependencies while still supporting decentralized execution.

Decentralized Training and Decentralized Execution In the DTDE paradigm, each agent learns independently and treats the other agents as part of the environment. We utilize IQL.
IQL. In Independent Q-Learning (IQL) [32], each agent learns a decentralized action-value function conditioned only on its own state or observation. Each agent updates its Q-network independently using a standard Q-learning objective [22], without explicitly modeling the actions or policies of other agents.

C.1 COMA

We additionally evaluate COMA [12] as an exploratory actor-critic baseline. COMA follows the centralized training with decentralized execution (CTDE) paradigm, using a centralized critic to train decentralized stochastic policies. Its critic estimates a counterfactual advantage for each agent by comparing the value of the agent's selected action to a baseline that marginalizes over that agent's alternative actions while holding the other agents' actions fixed. This counterfactual baseline is designed to address multi-agent credit assignment. Unlike the value-based methods emphasized in the main benchmark, COMA is on-policy and optimizes stochastic policies, so we report its results as exploratory and leave a broader evaluation of actor-critic methods to future work. COMA results are reported in Appendix K.

D Implementation and Hyperparameter Tuning

Implementation We implement VDN, QMIX, QTRAN, and IQL using PyMARL [28], while FDQN follows implementation in [7], and DQN is implemented as its non-factorized centralized counterpart.

Hyperparameter Tuning We perform a controlled, lightweight hyperparameter search over two optimization parameters: learning rate and ϵ-decay fraction. For each algorithm, we evaluate learning rates of 3×10^{-4} and 10^{-3}, and ϵ-decay fractions of 0.2 and 0.4. Hyperparameter tuning is performed on three representative problem scales: Small, Medium, and Large. For each algorithm and

Table 4: Selected hyperparameters across representative benchmark settings. Small, Medium, and Large denote representative tuning settings used to select hyperparameters for the full benchmark suite. LR denotes learning rate, and ϵ-decay denotes the fraction of training over which ϵ is decayed.

Algorithm	Small		Medium		Large	
	LR	ϵ-decay	LR	ϵ-decay	LR	ϵ-decay
DQN	0.0003	0.2	–	–	–	–
FDQN	0.0003	0.2	0.001	0.2	0.0003	0.4
VDN	0.001	0.2	0.001	0.2	0.001	0.2
QMIX	0.001	0.2	0.001	0.4	0.001	0.4
QTRAN	0.001	0.2	0.001	0.4	0.001	0.4
IQL	0.001	0.2	0.001	0.4	0.001	0.2

representative setting, we select the hyperparameter configuration that maximizes mean evaluation return over the final 20% of evaluation checkpoints, averaged across training seeds. This criterion emphasizes stable late-stage evaluation performance rather than transient peaks during training. When multiple configurations produce similar final return, we break ties using average return over the full training trajectory, favoring stability and sample efficiency.

The hyperparameters selected from the Small setting are used for experiments with 3 agents, those selected from the Medium setting are used for experiments with 5 agents and environment sizes smaller than 50×30, and those selected from the Large setting are used for experiments with environment size 50×30. For DQN, it was only computationally feasible to run the first three experiments with 3 agents. This strategy adapts hyperparameters to broad changes in problem scale while avoiding per-configuration overfitting and maintaining a consistent evaluation protocol across the benchmark.

E Full Learning Curves

For each STAT configuration, we plot evaluation return together with process-level diagnostics over training. Baseline experiments are in Figure 6 and Extreme experiments are in Figure 7. These curves show how performance, redundant assignment, and allocation breadth evolve as learning progresses. As Extreme experiments

F Full Per-Environment Results

We report the full per-environment numerical results supporting the main benchmark analysis. These tables report the absolute final or peak performance of each method in every STAT configuration. Table 5 reports final return, Table 6 reports maximum return and the timestep at which it is attained, Table 7 reports final conflict rate, Table 8 reports conflicts per task, Table 9 reports per-agent assignment diversity, and Table 10 reports task throughput.

For each table, values are reported as mean $\pm$ 95% confidence interval over five seeds. Bold indicates the best method for a configuration, and † indicates methods that are not statistically significantly different from the best according to Welch's two-sample t-test at $\alpha = 0.05$ [34]. For return, maximum return, per-agent assignment diversity, and task throughput, higher values are better. For conflict rate and conflicts per task, lower values are better.

Together, these tables show that outcome-level performance and coordination behavior do not always move together. Methods with similar final or maximum return can differ substantially in conflict rate, conflicts per task, and assignment diversity, especially in larger task-allocation settings. This supports the notion that return-based comparisons are incomplete without process-level diagnostics.

Figure 9 provides a different compact visual summary of the full benchmark. The heatmaps are intended as an overview of absolute metric values across methods and configurations. They show that return, conflict rate, per-agent assignment diversity, and throughput capture different aspects of behavior. In particular, methods that are close in return can still differ substantially in conflict rate and assignment diversity, reinforcing the need for process-level diagnostics.

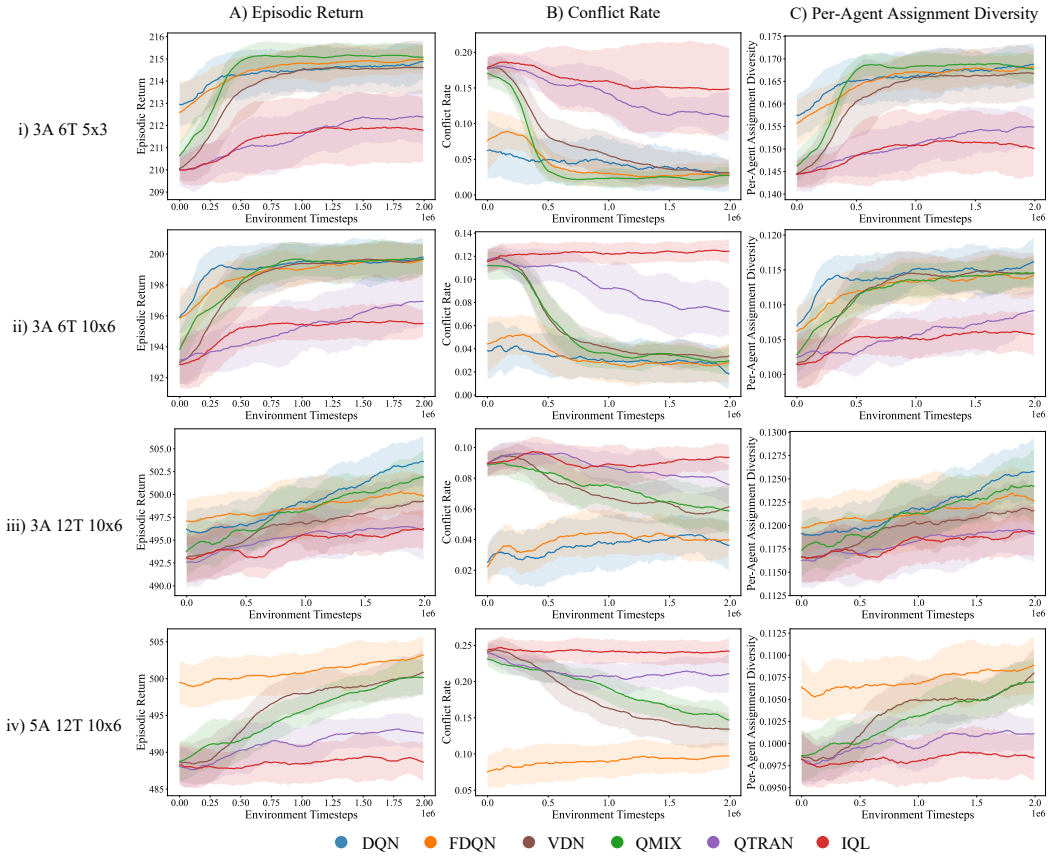


Figure 6: Learning curves for Baseline STAT configurations. Curves show mean evaluation performance across five seeds, with shaded regions denoting 95% confidence intervals. We report episodic return as the outcome-level metric and conflict rate, and per-agent assignment diversity as process-level diagnostics.

Table 5: Final return over 5 seeds, reported as mean ± 95% CI. Higher is better. Bold indicates the best method and † indicates methods not significantly different from the best at $\alpha = 0.05$.

Training Paradigm	**CTCE**		**CTDE**			**DTDE**
Environments/Methods	**DQN**	**FDQN**	**VDN**	**QMIX**	**QTRAN**	**IQL**
3A-6T-5x3	215.05 ± 0.7†	214.97 ± 0.7†	214.59 ± 1.1†	**215.06 ± 0.4**	212.31 ± 1.0	211.73 ± 1.4
3A-6T-10x6	**199.94 ± 1.4**	199.87 ± 0.7†	199.76 ± 0.8†	199.61 ± 0.8†	196.90 ± 1.5	195.52 ± 0.7
3A-12T-10x6	**504.20 ± 2.5**	499.72 ± 2.2	499.14 ± 2.6	502.07 ± 3.7†	495.87 ± 3.4	496.29 ± 1.9
5A-12T-10x6	–	**503.68 ± 2.5**	501.31 ± 3.2†	500.11 ± 2.2	492.66 ± 1.8	487.81 ± 3.0
5A-25T-25x15	–	1263.87 ± 14.2†	**1266.00 ± 5.3**	1261.98 ± 8.5†	1248.17 ± 7.9	1245.52 ± 5.4
5A-25T-50x30	–	857.58 ± 32.5†	**866.90 ± 26.3**	856.06 ± 17.1†	853.16 ± 16.9†	854.01 ± 9.0†
5A-50T-50x30	–	**2697.84 ± 19.6**	2666.88 ± 46.6†	2664.23 ± 66.7†	2648.79 ± 52.4†	2674.91 ± 35.8†
5A-100T-50x30	–	6075.01 ± 198.8†	5892.01 ± 185.1†	5934.09 ± 207.3†	**6144.00 ± 268.8**	6122.33 ± 182.5†
9A-25T-50x30	–	**909.51 ± 28.6**	909.08 ± 21.1†	902.58 ± 23.6†	888.71 ± 25.6†	872.16 ± 11.6

G Scaling Significance Tests

Table 14 (sideways, at the end) reports statistical tests for the controlled scaling comparisons shown in Figure 4. For each method, comparison, and metric, we test whether the change between configurations is statistically significant using Welch's two-sample t-test at $\alpha = 0.05$ [34]. The table reports the direction of change and whether the effect is significant. The purpose of this table is to provide statistical support for the scaling-induced changes discussed in the main paper (Section 5.4). Significant differences indicate that the observed effects are consistent across seeds rather than being driven only by random variation, supporting the interpretation that scaling changes coordination behavior and overall performance in measurable ways.

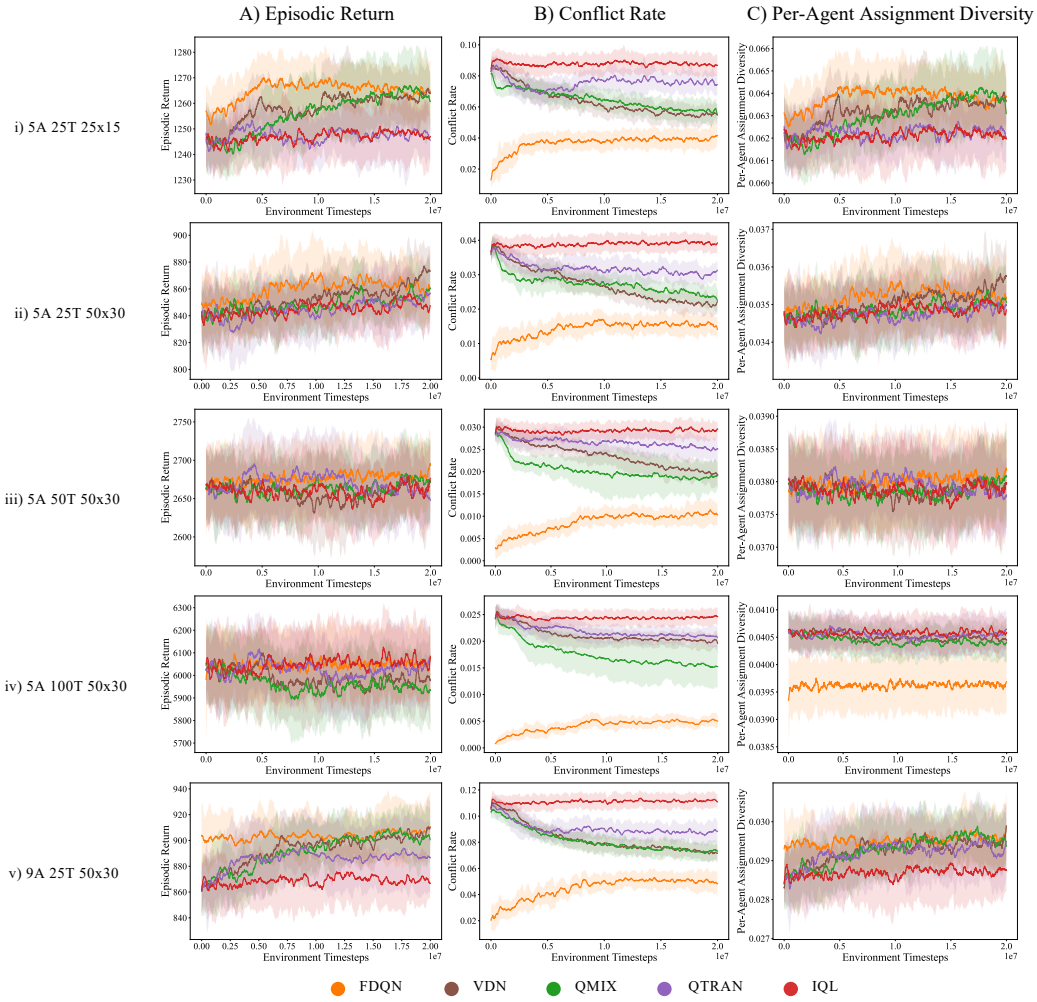


Figure 7: Learning curves for Extreme STAT configurations. Curves show mean evaluation performance across five seeds, with shaded regions denoting 95% confidence intervals. Return summarizes task performance, while conflict rate and per-agent assignment diversity summarize coordination behavior over training.

H Computational Efficiency

We report computational efficiency as an important context for interpreting scalability. A method that reduces coordination failures may still be impractical if it scales poorly with the nominal joint action space. Conversely, a method may remain computationally tractable while still producing poor coordination. We therefore report both wall-clock training time and training throughput.

Table 11 reports wall-clock training time in hours, and Table 12 reports average environment timesteps per second. Standard centralized DQN is evaluated only in the three smallest configurations because it explicitly represents values over the full joint action space. Although DQN is tractable in the smallest settings, its throughput drops sharply as the centralized joint-action output grows, making it infeasible for larger configurations. FDQN avoids this failure mode through a factorized centralized action representation and remains tractable across the full benchmark. It achieves the lowest wall-clock training time in all configurations where it is evaluated, often requiring less than half the training time of the CTDE and DTDE methods in the Extreme settings. This indicates that factorized centralized representations can provide practical scalability benefits in this testbed.

The PyMARL-based CTDE and DTDE methods have broadly similar computational profiles. Their throughput remains relatively stable across many configurations, but wall-clock time increases

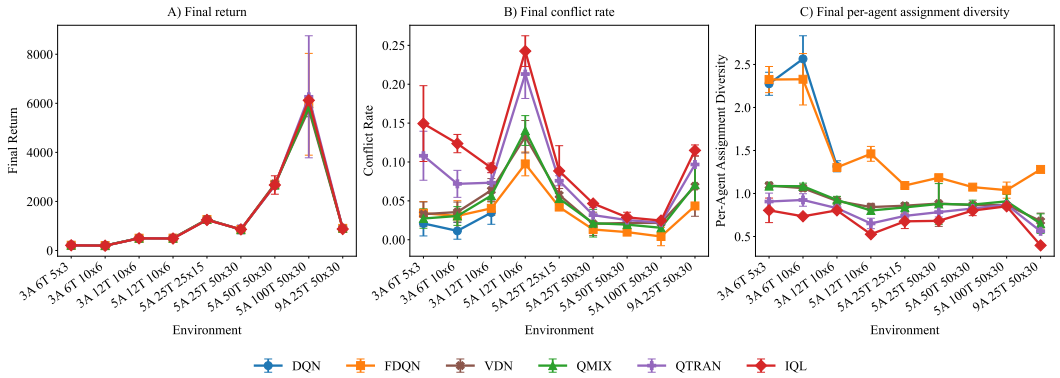


Figure 8: Full benchmark overview across STAT configurations. Final return summarizes task performance, while conflict rate and per-agent assignment diversity summarize coordination behavior. Methods with similar return can exhibit substantially different conflict rates and assignment diversity.

Table 6: Maximum return over 5 seeds, reported as mean ± 95% CI. Higher is better. Bold indicates the best method and † indicates methods not significantly different from the best at $\alpha = 0.05$. Peak Step denotes the median earliest evaluation timestep at which a seed attains its maximum reward.

Training Paradigm	**DTDE**		**CTDE**						**CTCE**			
Methods	**IQL**		**VDN**		**QMIX**		**QTRAN**		**FDQN**		**DQN**	
Environments/Metrics	**Max Reward**	**Peak Step**	**Max Reward**	**Peak Step**	**Max Reward**	**Peak Step**	**Max Reward**	**Peak Step**	**Max Reward**	**Peak Step**	**Max Reward**	**Peak Step**
3A-6T-5x3	**215.79 ± 0.5**	1.21M	215.59 ± 0.6†	1.92M	215.32 ± 1.0†	1.05M	215.70 ± 0.6†	1.56M	213.61 ± 0.7	1.60M	212.73 ± 1.6	1.52M
3A-6T-10x6	**201.02 ± 1.1**	1.81M	200.86 ± 1.1†	1.60M	200.53 ± 0.7†	1.66M	200.62 ± 0.9†	1.68M	198.53 ± 1.3	1.69M	196.99 ± 1.1	831k
3A-12T-10x6	**507.05 ± 2.2**	1.87M	503.83 ± 1.3†	1.27M	502.22 ± 3.0†	1.52M	504.24 ± 1.5†	1.97M	498.92 ± 3.0	1.52M	500.54 ± 0.8	1.24M
5A-12T-10x6	–	–	**505.74 ± 2.1**	1.80M	503.09 ± 1.6†	1.75M	502.72 ± 1.5†	1.89M	496.26 ± 1.0	1.50M	493.13 ± 1.8	871k
5A-25T-25x15	–	–	**1289.77 ± 6.9**	10.65M	1288.83 ± 9.1†	12.50M	1285.16 ± 11.5†	16.14M	1271.63 ± 7.9†	3.77M	1271.69 ± 7.1	3.04M
5A-25T-50x30	–	–	**911.41 ± 16.8**	6.91M	906.98 ± 15.9†	18.44M	906.91 ± 11.1†	7.93M	887.67 ± 21.9	11.83M	899.48 ± 15.5†	10.13M
5A-50T-50x30	–	–	**2818.90 ± 17.1**	8.83M	2785.10 ± 44.9†	5.02M	2789.98 ± 40.5†	14.95M	2784.33 ± 45.3†	3.91M	2791.03 ± 35.6	17.78M
5A-100T-50x30	–	–	**6521.22 ± 71.3**	12.21M	6510.74 ± 80.6†	12.88M	6442.63 ± 49.6†	1.99M	6441.12 ± 86.1†	4.23M	6497.27 ± 122.1†	11.61M
9A-25T-50x30	–	–	**941.14 ± 19.0**	11.49M	937.69 ± 13.4†	15.99M	938.90 ± 18.1†	17.08M	925.98 ± 20.6†	8.13M	911.26 ± 17.6	7.78M

substantially in the Extreme regime due to the larger training budget and more expensive environment dynamics.

I Additional Method-Level Observations

Our main analysis focuses on scaling-induced changes. Here, we summarize additional method-level patterns observed across the full set of STAT configurations. These observations are intended to provide additional context for the main results.

Centralized Training Centralized Execution Methods. The centralized methods illustrate the trade-off between joint-action reasoning and computational tractability. Standard DQN performs competitively in the smallest configurations, where explicit centralized joint-action reasoning remains feasible. However, it becomes infeasible beyond the three smallest settings because its output layer enumerates the full joint action space. FDQN avoids this failure mode through a factorized centralized representation and remains tractable across the full benchmark. Across the full results, FDQN often achieves low conflict rates and strong task performance, suggesting that structured centralized representations can reduce redundant assignment when the joint-action representation remains scalable.

Centralized Training Decentralized Execution Methods. The CTDE methods, including VDN, QMIX, and QTRAN, remain tractable across all configurations and generally achieve competitive return under scale. This makes them useful references for studying the coordination–scalability trade-off. However, their process-level diagnostics reveal that comparable return does not necessarily imply comparable coordination quality. In several configurations, CTDE methods remain competitive in return while exhibiting higher conflict rates or lower per-agent assignment diversity than the strongest centralized or factorized approaches. This supports the notion that return-based comparisons alone can obscure differences in redundant assignment and allocation behavior.

Table 7: Final conflict rate over 5 seeds, reported as mean ± 95% CI. Lower is better. Bold indicates the best method and † indicates methods not significantly different from the best at $\alpha = 0.05$.

Training Paradigm	**CTCE**		**CTDE**			**DTDE**
Environments/Methods	**DQN**	**FDQN**	**VDN**	**QMIX**	**QTRAN**	**IQL**
3A-6T-5x3	**0.0210 ± 0.0161**	0.0341 ± 0.0151†	0.0329 ± 0.0156†	0.0273 ± 0.0125†	0.1080 ± 0.0315	0.1494 ± 0.0488
3A-6T-10x6	**0.0114 ± 0.0107**	0.0307 ± 0.0195†	0.0356 ± 0.0128	0.0305 ± 0.0121	0.0719 ± 0.0174	0.1237 ± 0.0118
3A-12T-10x6	**0.0349 ± 0.0150**	0.0404 ± 0.0090†	0.0637 ± 0.0150	0.0568 ± 0.0142	0.0733 ± 0.0132	0.0924 ± 0.0062
5A-12T-10x6	–	**0.0975 ± 0.0153**	0.1325 ± 0.0209	0.1405 ± 0.0192	0.2131 ± 0.0314	0.2426 ± 0.0197
5A-25T-25x15	–	**0.0420 ± 0.0023**	0.0567 ± 0.0053	0.0518 ± 0.0060	0.0773 ± 0.0033	0.0859 ± 0.0045
5A-25T-50x30	–	**0.0132 ± 0.0038**	0.0214 ± 0.0033	0.0223 ± 0.0026	0.0314 ± 0.0014	0.0412 ± 0.0046
5A-50T-50x30	–	**0.0096 ± 0.0011**	0.0189 ± 0.0034	0.0191 ± 0.0021	0.0253 ± 0.0036	0.0295 ± 0.0012
5A-100T-50x30	–	**0.0048 ± 0.0011**	0.0200 ± 0.0020	0.0155 ± 0.0043	0.0206 ± 0.0024	0.0246 ± 0.0020
9A-25T-50x30	–	**0.0465 ± 0.0124**	0.0715 ± 0.0051	0.0699 ± 0.0107	0.0889 ± 0.0073	0.1138 ± 0.0056

Table 8: Final conflicts per task over 5 seeds, reported as mean ± 95% CI. Lower is better. Bold indicates the best method and † indicates methods not significantly different from the best at $\alpha = 0.05$.

Training Paradigm	**CTCE**		**CTDE**			**DTDE**
Environments/Methods	**DQN**	**FDQN**	**VDN**	**QMIX**	**QTRAN**	**IQL**
3A-6T-5x3	**0.0417 ± 0.0327**	0.0683 ± 0.0322†	0.0667 ± 0.0335†	0.0550 ± 0.0260†	0.2367 ± 0.0704	0.3367 ± 0.1201
3A-6T-10x6	**0.0333 ± 0.0319**	0.0900 ± 0.0573†	0.1050 ± 0.0391	0.0900 ± 0.0369	0.2233 ± 0.0564	0.3950 ± 0.0324
3A-12T-10x6	**0.0925 ± 0.0383**	0.1108 ± 0.0244†	0.1767 ± 0.0418	0.1542 ± 0.0396	0.2075 ± 0.0336	0.2608 ± 0.0209
5A-12T-10x6	–	**0.1808 ± 0.0326**	0.2458 ± 0.0425	0.2650 ± 0.0376	0.4233 ± 0.0586	0.4992 ± 0.0270
5A-25T-25x15	–	**0.1324 ± 0.0101**	0.1780 ± 0.0167	0.1652 ± 0.0185	0.2504 ± 0.0107	0.2800 ± 0.0139
5A-25T-50x30	–	**0.0760 ± 0.0232**	0.1216 ± 0.0156	0.1284 ± 0.0166	0.1812 ± 0.0091	0.2368 ± 0.0251
5A-50T-50x30	–	**0.0504 ± 0.0054**	0.1000 ± 0.0178	0.1010 ± 0.0120	0.1340 ± 0.0184	0.1558 ± 0.0083
5A-100T-50x30	–	**0.0242 ± 0.0055**	0.0987 ± 0.0095	0.0765 ± 0.0208	0.1004 ± 0.0105	0.1209 ± 0.0098
9A-25T-50x30	–	**0.1756 ± 0.0493**	0.2672 ± 0.0155	0.2672 ± 0.0384	0.3360 ± 0.0248	0.4408 ± 0.0144

Decentralized Training Decentralized Execution Methods. IQL provides a decentralized baseline for testing how independent learning behaves under assignment interdependence. Across many configurations, IQL exhibits higher conflict rates than the more centralized or CTDE methods, indicating that independent learners are more vulnerable to overlapping task selections when assignment decisions are coupled across agents. In some high task-to-agent-ratio settings, IQL remains competitive in return because many reward opportunities are available, but its conflict metrics indicate that this performance can coexist with poorer coordination. This again illustrates why process-level diagnostics are needed alongside aggregate return.

Overall, these method-level observations complement the controlled scaling analysis in Section 5.4. The results suggest that method structure affects how coordination failures appear under scale. Explicit or factorized centralized reasoning can reduce redundant assignment when tractable, CTDE methods remain scalable and competitive across configurations, and independent learning is more vulnerable to redundant assignment when task choices are strongly interdependent.

J Additional Process Diagnostics

Our main set of process-level diagnostics (Section 4.2) are directly interpretable across scaling axes: return, conflict rate, conflicts per task, per-agent assignment diversity, and throughput. Here, we report additional mechanism-level diagnostics that provide finer detail about how coordination failures arise within STAT's commitment-constrained decision structure.

These diagnostics are useful because assignment decisions in STAT occur only at sparse decision points, separated by movement and execution phases. As a result, changes in raw conflict or assignment diversity can arise either because coordination quality changes, or simply because the number of agents simultaneously available to make assignment decisions changes. The metrics below help disentangle these effects.

Forced idle rate measures the agent-level cost of conflict resolution. When multiple agents select the same task, one agent retains the assignment and the others are forced to idle for that timestep. We compute this as the number of forced-idle agents per episode timestep, providing a measure of how redundant assignments reduce usable team capacity.

Decision-active agent fraction measures the average fraction of agents that are at meaningful assignment decision points rather than committed to deterministic movement or task execution.

Table 9: Final per-agent assignment diversity over 5 seeds, reported as mean ± 95% CI. Higher is better. Bold indicates the best method and † indicates methods not significantly different from the best at $\alpha = 0.05$.

Training Paradigm	CTCE		CTDE			DTDE
Environments/Methods	DQN	FDQN	VDN	QMIX	QTRAN	IQL
3A-6T-5x3	$2.275 \pm 0.134^{\dagger}$	**2.324 ± 0.152**	1.092 ± 0.034	1.086 ± 0.044	0.906 ± 0.098	0.804 ± 0.143
3A-6T-10x6	**2.565 ± 0.269**	$2.328 \pm 0.299^{\dagger}$	1.062 ± 0.036	1.084 ± 0.036	0.925 ± 0.073	0.735 ± 0.014
3A-12T-10x6	**1.316 ± 0.064**	$1.303 \pm 0.042^{\dagger}$	0.913 ± 0.049	0.925 ± 0.042	0.829 ± 0.026	0.804 ± 0.023
5A-12T-10x6	–	**1.461 ± 0.086**	0.842 ± 0.043	0.802 ± 0.038	0.652 ± 0.061	0.526 ± 0.008
5A-25T-25x15	–	**1.093 ± 0.023**	0.859 ± 0.014	0.837 ± 0.037	0.732 ± 0.021	0.681 ± 0.009
5A-25T-50x30	–	**1.183 ± 0.031**	0.890 ± 0.028	0.855 ± 0.038	0.776 ± 0.009	0.699 ± 0.016
5A-50T-50x30	–	**1.070 ± 0.008**	0.886 ± 0.028	0.881 ± 0.030	0.825 ± 0.023	0.799 ± 0.007
5A-100T-50x30	–	**1.036 ± 0.007**	0.875 ± 0.010	0.907 ± 0.033	0.873 ± 0.011	0.851 ± 0.008
9A-25T-50x30	–	**1.222 ± 0.126**	0.668 ± 0.036	0.663 ± 0.055	0.554 ± 0.033	0.397 ± 0.004

Table 10: Final task throughput over 5 seeds, reported as mean ± 95% CI. Higher is better. Bold indicates the best method and † indicates methods not significantly different from the best at $\alpha = 0.05$.

Training Paradigm	CTCE		CTDE			DTDE
Environments/Methods	DQN	FDQN	VDN	QMIX	QTRAN	IQL
3A-6T-5x3	**0.5052 ± 0.0128**	$0.5015 \pm 0.0148^{\dagger}$	$0.4965 \pm 0.0149^{\dagger}$	$0.4981 \pm 0.0111^{\dagger}$	0.4568 ± 0.0112	0.4468 ± 0.0170
3A-6T-10x6	**0.3467 ± 0.0133**	$0.3404 \pm 0.0067^{\dagger}$	$0.3399 \pm 0.0082^{\dagger}$	$0.3400 \pm 0.0071^{\dagger}$	0.3224 ± 0.0109	0.3130 ± 0.0097
3A-12T-10x6	**0.3760 ± 0.0103**	0.3643 ± 0.0082	0.3605 ± 0.0055	$0.3693 \pm 0.0150^{\dagger}$	0.3529 ± 0.0099	0.3545 ± 0.0068
5A-12T-10x6	–	**0.5406 ± 0.0194**	$0.5398 \pm 0.0144^{\dagger}$	$0.5305 \pm 0.0150^{\dagger}$	0.5030 ± 0.0086	0.4857 ± 0.0154
5A-25T-25x15	–	$0.3176 \pm 0.0078^{\dagger}$	**0.3183 ± 0.0030**	$0.3137 \pm 0.0053^{\dagger}$	0.3087 ± 0.0053	0.3067 ± 0.0037
5A-25T-50x30	–	$0.1741 \pm 0.0069^{\dagger}$	**0.1754 ± 0.0063**	$0.1738 \pm 0.0028^{\dagger}$	$0.1733 \pm 0.0019^{\dagger}$	$0.1738 \pm 0.0038^{\dagger}$
5A-50T-50x30	–	**0.1899 ± 0.0019**	$0.1890 \pm 0.0029^{\dagger}$	$0.1888 \pm 0.0036^{\dagger}$	$0.1886 \pm 0.0029^{\dagger}$	$0.1894 \pm 0.0035^{\dagger}$
5A-100T-50x30	–	0.1982 ± 0.0033	$0.2024 \pm 0.0013^{\dagger}$	$0.2025 \pm 0.0015^{\dagger}$	**0.2048 ± 0.0025**	$0.2037 \pm 0.0013^{\dagger}$
9A-25T-50x30	–	$0.2656 \pm 0.0093^{\dagger}$	**0.2676 ± 0.0064**	$0.2615 \pm 0.0045^{\dagger}$	$0.2646 \pm 0.0093^{\dagger}$	0.2582 ± 0.0054

Conflicts per decision opportunity measures conflicts relative to the amount of assignment decision activity in an episode. We define decision opportunities as the average number of decision-active agents multiplied by episode length, and divide total conflicts by this quantity.

Assignment diversity per decision-active agent measures how many distinct task assignments are produced per decision-active agent, normalizing assignment diversity by the number of agents actually available to make assignment decisions.

Figure 10 provides mechanism-level context for the main scaling results. Under **environment-size scaling**, the most consistent pattern is a decrease in decision-active agent fraction across methods, for both the baseline and extreme environment-size comparisons. This indicates that, as the environment becomes larger while the number of agents and tasks is held fixed, agents spend a smaller fraction of episode time at assignment decision points and a larger fraction of time committed to movement or task execution. Forced idle rate generally decreases or remains close to zero, suggesting that larger environments do not increase the agent-level cost of conflict resolution and may reduce direct assignment contention for several methods. Changes in conflicts per decision opportunity are comparatively small and mixed, with uncertainty intervals often overlapping zero, indicating limited evidence that spatial scaling substantially worsens coordination quality per available decision opportunity. Assignment diversity per decision-active agent is also mostly stable, aside from a larger positive change for DQN in the Baseline comparison, suggesting that the main effect of environment-size scaling is reduced opportunity for reassignment rather than a broad collapse in assignment quality once agents become decision-active.

Under **task scaling**, the diagnostics show that adding tasks changes both assignment availability and per-opportunity allocation behavior. In the Baseline comparison, decision-active agent fraction increases most strongly for DQN and FDQN, with smaller or near-zero changes for several CTDE methods and IQL. This suggests that additional tasks can keep some agents at assignment decision points more often, but the effect is method-dependent rather than uniform. At the same time, assignment diversity per decision-active agent drops sharply for DQN and FDQN in the Baseline comparison, indicating that more decision activity does not necessarily translate into more distinct assignments per active decision-maker. For the larger task-scaling comparisons, changes in assignment diversity are much smaller and often near zero. Conflicts per decision opportunity are mixed across methods and scaling regimes, with several uncertainty intervals overlapping zero. Forced idle rate also varies by method, increasing for some methods and decreasing for others, with the clearest

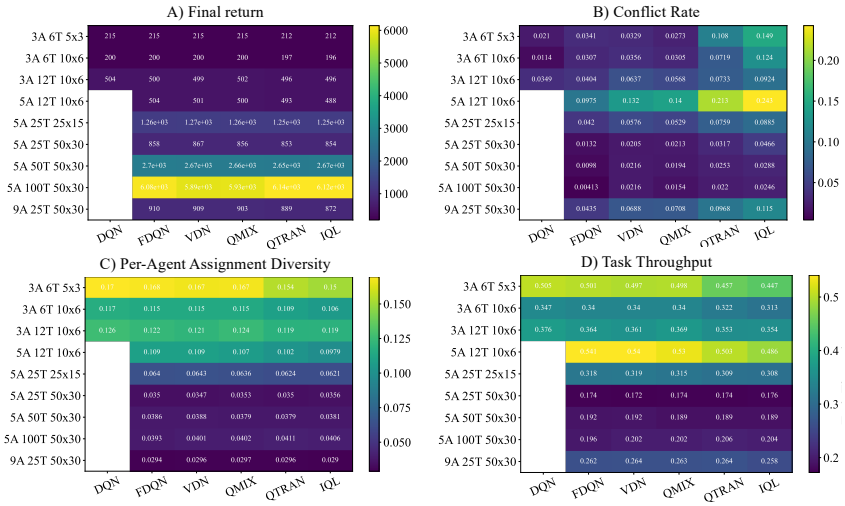


Figure 9: Full benchmark overview across STAT configurations. Each heatmap reports the final evaluation metric averaged over five seeds. Final return summarizes task performance, conflict rate measures redundant assignment frequency, per-agent assignment diversity measures allocation breadth, and task throughput measures completion efficiency.

Table 11: Wall-clock training time across STAT configurations, reported in hours as mean ± 95% CI over five seeds. Lower is better. – indicates that the method was not evaluated because it was computationally infeasible.

Training Paradigm	**CTCE**		**CTDE**			**DTDE**
Environments/Methods	**DQN**	**FDQN**	**VDN**	**QMIX**	**QTRAN**	**IQL**
3A-6T-5x3	1.87 ± 0.15	0.91 ± 0.02	2.93 ± 1.09	2.62 ± 0.22	2.74 ± 0.64	2.33 ± 0.04
3A-6T-10x6	1.84 ± 0.12	1.00 ± 0.09	2.70 ± 0.72	2.25 ± 0.04	2.42 ± 0.42	2.27 ± 0.12
3A-12T-10x6	3.45 ± 0.03	0.86 ± 0.03	2.10 ± 0.20	2.07 ± 0.09	2.21 ± 0.44	2.07 ± 0.10
5A-12T-10x6	–	1.09 ± 0.02	2.21 ± 0.07	2.41 ± 0.11	2.31 ± 0.05	2.54 ± 0.52
5A-25T-25x15	–	11.22 ± 0.46	29.42 ± 0.68	30.69 ± 2.26	30.70 ± 0.97	29.44 ± 0.57
5A-25T-50x30	–	11.76 ± 0.41	28.92 ± 1.48	28.91 ± 0.43	29.33 ± 0.57	28.94 ± 0.87
5A-50T-50x30	–	13.68 ± 0.13	32.25 ± 0.48	32.54 ± 0.40	32.80 ± 0.49	32.20 ± 0.49
5A-100T-50x30	–	17.77 ± 0.11	41.73 ± 0.70	41.91 ± 0.53	42.03 ± 0.69	41.43 ± 0.70
9A-25T-50x30	–	16.01 ± 0.23	33.74 ± 2.85	33.69 ± 0.96	33.38 ± 0.71	33.13 ± 0.70

decrease appearing for IQL in the larger task-scaling comparisons. Overall, task scaling expands return and assignment opportunities, but these additional opportunities do not uniformly improve per-decision coordination. Instead, results suggest that the effect of adding tasks depends strongly on the learning method and on whether scaling occurs in the smaller Baseline regime or the larger Extreme regimes.

Under **agent scaling**, the mechanism-level diagnostics show the clearest evidence of coordination stress. Forced idle rate increases for all methods in both the Baseline and Extreme comparisons, with especially large increases for QTRAN and IQL. This indicates that adding agents creates more overlapping assignment attempts and more agents losing conflict resolution. Conflicts per decision opportunity also generally increase, suggesting that the rise in conflict is not only a byproduct of having more agents, but also reflects greater contention per unit of decision activity. Decision-active agent fraction decreases slightly for FDQN, is near zero for VDN and QMIX, and increases most clearly for QTRAN and IQL. Thus, adding agents does not uniformly change the fraction of agents at meaningful assignment points. Assignment diversity per decision-active agent tends to decline for most methods, especially in the extreme comparison, showing that added decision capacity is not converted proportionally into distinct assignments. Together, these patterns support the main conclusion that increasing the number of agents creates the strongest coordination pressure and that additional team capacity is beneficial only when methods can translate it into distinct work.

Table 12: Training throughput across STAT configurations, reported as environment timesteps per second and averaged over five seeds with 95% CI. Higher is better. – indicates that the method was not evaluated.

Training Paradigm	**CTCE**		**CTDE**			**DTDE**
Environments/Methods	**DQN**	**FDQN**	**VDN**	**QMIX**	**QTRAN**	**IQL**
3A-6T-5x3	304.95 ± 8.4	651.25 ± 1.8	585.28 ± 12.8	588.79 ± 6.5	591.51 ± 10.3	581.50 ± 15.0
3A-6T-10x6	326.63 ± 3.1	698.67 ± 62.0	594.15 ± 23.6	603.12 ± 5.7	599.77 ± 8.9	584.02 ± 26.4
3A-12T-10x6	164.04 ± 1.9	708.92 ± 17.9	595.63 ± 30.4	605.16 ± 6.3	592.36 ± 22.3	590.13 ± 26.3
5A-12T-10x6	–	556.31 ± 20.3	575.19 ± 7.8	573.94 ± 4.0	569.53 ± 9.8	566.40 ± 23.3
5A-25T-25x15	–	526.52 ± 22.1	548.01 ± 4.3	540.53 ± 11.4	539.69 ± 6.8	545.24 ± 5.3
5A-25T-50x30	–	515.07 ± 18.5	541.62 ± 14.6	538.73 ± 7.6	539.68 ± 9.3	539.20 ± 8.1
5A-50T-50x30	–	473.66 ± 6.8	472.20 ± 7.0	469.93 ± 2.1	472.72 ± 6.6	471.41 ± 7.5
5A-100T-50x30	–	421.39 ± 2.8	375.49 ± 7.0	375.01 ± 5.7	377.34 ± 7.0	376.93 ± 6.3
9A-25T-50x30	–	369.71 ± 5.7	486.53 ± 11.0	484.76 ± 5.1	487.88 ± 3.0	485.53 ± 6.5

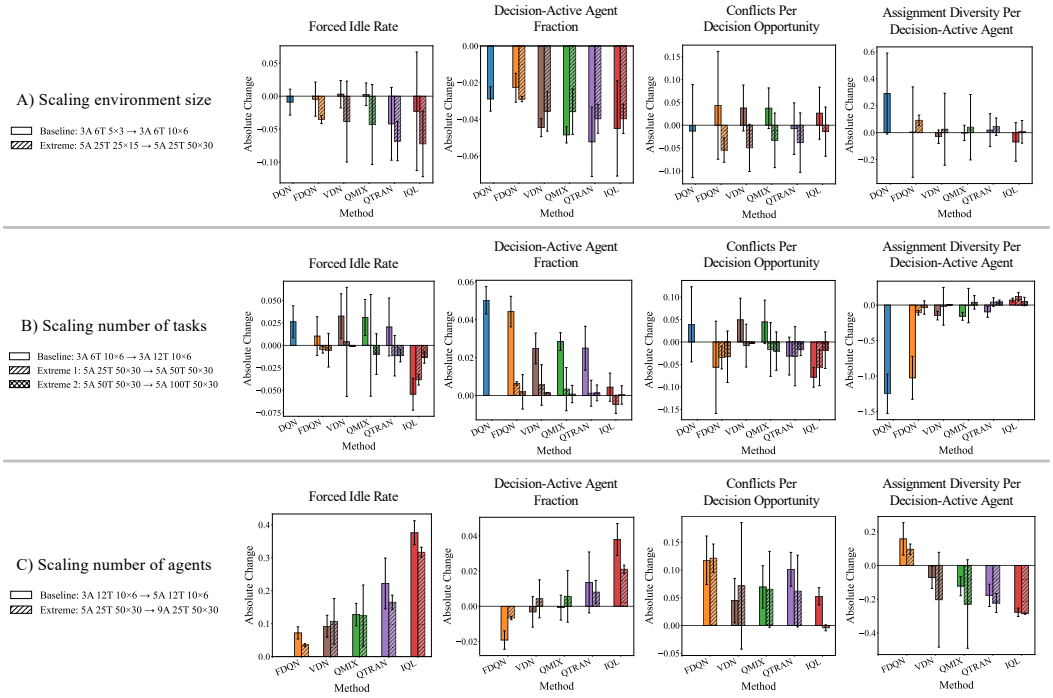


Figure 10: Additional mechanism-level scaling diagnostics. Each row isolates one controlled scaling axis: (A) environment size, (B) number of tasks, and (C) number of agents. Bars show mean change across five seeds with 95% confidence intervals. These metrics provide a more detailed view of coordination behavior by accounting for agent availability and decision opportunity.

K Exploratory COMA Results

We additionally report exploratory COMA results using the same STAT evaluation protocol as the main benchmark. Table 13 shows the final metrics, including return, conflict rate, conflicts per task, per-agent assignment diversity, and task throughput. COMA is an on-policy actor-critic method, whereas the main benchmark focuses on value-based methods trained with replay. Because COMA differs substantially in optimization procedure, exploration behavior, and hyperparameter sensitivity, we treat these results as an initial actor-critic comparison rather than a definitive evaluation of policy-gradient MARL methods.

This distinction is especially important in STAT because assignment decisions occur only at sparse, high-impact decision points. Action masking and finite-state commitment create intervals in which agents have limited meaningful choices, which may reduce the frequency of informative policy-gradient updates for assignment coordination. As a result, COMA may require different tuning choices, longer training budgets, or alternative actor-critic implementations to be fully competitive.

The COMA results are included to broaden the empirical context of the benchmark, while the main conclusions are drawn from the value-based methods evaluated consistently across all STAT configurations. A more complete evaluation of actor-critic methods, including MAPPO [36] and other on-policy approaches, is left for future work.

Table 13: Exploratory COMA results across STAT configurations. Values are reported as mean $\pm$ 95% CI over five seeds. Final return and final task throughput are outcome and efficiency metrics, while final conflict rate, final conflicts per task, and final per-agent assignment diversity characterize coordination behavior.

Configuration	Return	Conflict Rate	Conflicts per Task	Per-Agent Assignment Diversity	Task Throughput
3A-6T-5x3	214.94 ± 0.44	0.0278 ± 0.0155	0.055 ± 0.031	0.169 ± 0.002	0.505 ± 0.007
3A-6T-10x6	198.66 ± 1.01	0.0562 ± 0.0305	0.170 ± 0.094	0.112 ± 0.003	0.331 ± 0.009
3A-12T-10x6	493.36 ± 2.99	0.0963 ± 0.0119	0.278 ± 0.036	0.117 ± 0.003	0.347 ± 0.009
5A-12T-10x6	488.20 ± 2.38	0.2510 ± 0.0154	0.521 ± 0.031	0.097 ± 0.001	0.482 ± 0.005
5A-25T-25x15	1248.35 ± 13.82	0.0904 ± 0.0082	0.292 ± 0.026	0.063 ± 0.002	0.310 ± 0.011
5A-25T-50x30	835.63 ± 9.23	0.0382 ± 0.0030	0.223 ± 0.017	0.035 ± 0.001	0.171 ± 0.003
5A-50T-50x30	2676.18 ± 34.02	0.0303 ± 0.0027	0.159 ± 0.011	0.038 ± 0.001	0.190 ± 0.004
5A-100T-50x30	6003.06 ± 193.40	0.0248 ± 0.0024	0.122 ± 0.011	0.040 ± 0.000	0.203 ± 0.001
9A-25T-50x30	853.51 ± 10.83	0.1050 ± 0.0055	0.418 ± 0.021	0.028 ± 0.000	0.251 ± 0.002

Table 14: Statistical tests for controlled scaling comparisons. Each cell shows the direction of change from the first configuration to the second configuration. ↑ indicates an increase, ↓ indicates a decrease, * indicates $p < 0.05$, and ns indicates not significant. "–" indicates that the comparison was not available.

Comparison	Method	R	Th	CR	TC	CPT	PAD
Env Size Baseline	DQN	$\downarrow^{*}$	$\downarrow^{*}$	$\downarrow^{ns}$	$\downarrow^{ns}$	$\downarrow^{ns}$	$\downarrow^{*}$
	FDQN	$\downarrow^{*}$	$\downarrow^{*}$	$\downarrow^{ns}$	$\uparrow^{ns}$	$\uparrow^{ns}$	$\downarrow^{*}$
	VDN	$\downarrow^{*}$	$\downarrow^{*}$	$\uparrow^{ns}$	$\uparrow^{ns}$	$\uparrow^{ns}$	$\downarrow^{*}$
	QMIX	$\downarrow^{*}$	$\downarrow^{*}$	$\uparrow^{ns}$	$\uparrow^{ns}$	$\uparrow^{ns}$	$\downarrow^{*}$
	QTRAN	$\downarrow^{*}$	$\downarrow^{*}$	$\downarrow^{*}$	$\downarrow^{ns}$	$\downarrow^{ns}$	$\downarrow^{*}$
	IQL	$\downarrow^{*}$	$\downarrow^{*}$	$\downarrow^{ns}$	$\uparrow^{ns}$	$\uparrow^{ns}$	$\downarrow^{*}$
Env Size Extreme	DQN	–	–	–	–	–	–
	FDQN	$\downarrow^{*}$	$\downarrow^{*}$	$\downarrow^{*}$	$\downarrow^{*}$	$\downarrow^{*}$	$\downarrow^{*}$
	VDN	$\downarrow^{*}$	$\downarrow^{*}$	$\downarrow^{*}$	$\downarrow^{*}$	$\downarrow^{*}$	$\downarrow^{*}$
	QMIX	$\downarrow^{*}$	$\downarrow^{*}$	$\downarrow^{*}$	$\downarrow^{*}$	$\downarrow^{*}$	$\downarrow^{*}$
	QTRAN	$\downarrow^{*}$	$\downarrow^{*}$	$\downarrow^{*}$	$\downarrow^{*}$	$\downarrow^{*}$	$\downarrow^{*}$
	IQL	$\downarrow^{*}$	$\downarrow^{*}$	$\downarrow^{*}$	$\downarrow^{*}$	$\downarrow^{*}$	$\downarrow^{*}$
Tasks Baseline	DQN	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$
	FDQN	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{ns}$	$\uparrow^{*}$	$\uparrow^{ns}$	$\uparrow^{*}$
	VDN	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$
	QMIX	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$
	QTRAN	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{ns}$	$\uparrow^{*}$	$\downarrow^{ns}$	$\uparrow^{*}$
	IQL	$\uparrow^{*}$	$\uparrow^{*}$	$\downarrow^{*}$	$\uparrow^{*}$	$\downarrow^{*}$	$\uparrow^{*}$
Tasks Extreme 1	DQN	–	–	–	–	–	–
	FDQN	$\uparrow^{*}$	$\uparrow^{*}$	$\downarrow^{ns}$	$\uparrow^{*}$	$\downarrow^{*}$	$\uparrow^{*}$
	VDN	$\uparrow^{*}$	$\uparrow^{*}$	$\downarrow^{ns}$	$\uparrow^{*}$	$\downarrow^{*}$	$\uparrow^{*}$
	QMIX	$\uparrow^{*}$	$\uparrow^{*}$	$\downarrow^{*}$	$\uparrow^{*}$	$\downarrow^{*}$	$\uparrow^{*}$
	QTRAN	$\uparrow^{*}$	$\uparrow^{*}$	$\downarrow^{*}$	$\uparrow^{*}$	$\downarrow^{*}$	$\uparrow^{*}$
	IQL	$\uparrow^{*}$	$\uparrow^{*}$	$\downarrow^{*}$	$\uparrow^{*}$	$\downarrow^{*}$	$\uparrow^{*}$
Tasks Extreme 2	DQN	–	–	–	–	–	–
	FDQN	$\uparrow^{*}$	$\uparrow^{*}$	$\downarrow^{*}$	$\downarrow^{ns}$	$\downarrow^{*}$	$\uparrow^{*}$
	VDN	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{ns}$	$\uparrow^{*}$	$\downarrow^{ns}$	$\uparrow^{*}$
	QMIX	$\uparrow^{*}$	$\uparrow^{*}$	$\downarrow^{ns}$	$\uparrow^{*}$	$\downarrow^{*}$	$\uparrow^{*}$
	QTRAN	$\uparrow^{*}$	$\uparrow^{*}$	$\downarrow^{*}$	$\uparrow^{*}$	$\downarrow^{*}$	$\uparrow^{*}$
	IQL	$\uparrow^{*}$	$\uparrow^{*}$	$\downarrow^{*}$	$\uparrow^{*}$	$\downarrow^{*}$	$\uparrow^{*}$
Agents Baseline	DQN	–	–	–	–	–	–
	FDQN	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\downarrow^{*}$
	VDN	$\uparrow^{ns}$	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\downarrow^{*}$
	QMIX	$\downarrow^{ns}$	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\downarrow^{*}$
	QTRAN	$\downarrow^{ns}$	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\downarrow^{*}$
	IQL	$\downarrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\downarrow^{*}$
Agents Extreme	DQN	–	–	–	–	–	–
	FDQN	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\downarrow^{*}$
	VDN	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\downarrow^{*}$
	QMIX	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\downarrow^{*}$
	QTRAN	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\downarrow^{*}$
	IQL	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\uparrow^{*}$	$\downarrow^{*}$

Abbreviations. R denotes return, Th denotes task completion throughput, CR denotes conflict rate, TC denotes total task assignment conflicts, CPT denotes conflicts per task, and PAD denotes per-agent assignment diversity. Env Size Baseline compares 3A-6T-5x3 to 3A-6T-10x6. Env Size Extreme compares 5A-25T-25x15 to 5A-25T-50x30. Tasks Baseline compares 3A-6T-10x6 to 3A-12T-10x6. Tasks Extreme 1 compares 5A-25T-50x30 to 5A-50T-50x30. Tasks Extreme 2 compares 5A-50T-50x30 to 5A-100T-50x30. Agents Baseline compares 3A-12T-10x6 to 5A-12T-10x6. Agents Extreme compares 5A-25T-50x30 to 9A-25T-50x30.