# TOPOLOGICAL COHERENCE FOR SELF-EVOLVING MULTI-AGENT SYSTEMS

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Complex tasks inherently couple workflow structure, agent responsibility, collaboration, and memory access: task regions delimit responsibility and tool scope, cross-region dependencies give rise to handoffs, and ownership boundaries delimit private and selectively shared memory. Existing methods can jointly optimize agent and communication structures, yet such optimization does not by itself require responsibility, handoff, and memory boundaries to remain consistent with task dependencies. We term this requirement *topological coherence*. We introduce TOCOMAS, a *Topology-Coherent Multi-Agent System*. TOCOMAS grounds a task graph in tool interfaces, organizes compatible task nodes into reusable responsibility domains, and derives dependency-induced and profile-conditioned collaboration together with boundary-regulated memory visibility. During online self-evolution, TOCOMAS proposes coupled changes to agent, collaboration, and memory policies, retaining for subsequent tasks only candidates that satisfy structural constraints and improve evaluated reward. Across BBEH, WorkBench, SWE-Bench-Verified, and CoMemBench, TOCOMAS improves task success over baselines across backbones. CoMemBench also shows gains over the self-evolving baseline in verified progress, handoffs, and memory isolation. Code is available at `https://anonymous.4open.science/r/ToCoMAS_opensource-463F`.

## 1 INTRODUCTION

Large language model (LLM) based multi-agent systems (MAS) provide a general framework for combining specialized reasoning, tool use, and interaction policies (Guo et al., 2024; Li et al., 2024). Conversational and role-based systems such as CAMEL, AutoGen, and AgentVerse (Li et al., 2023; Wu et al., 2024; Chen et al., 2024), together with peer review, debate, MetaGPT, and ChatDev (Xu et al., 2023; Du et al., 2024; Hong et al., 2024; Qian et al., 2024), make increasingly complex tasks accessible. This progress also turns organization into a central design problem: a MAS must decide which task regions each agent owns, which tools and information it may access, where verification is required, and how local outputs compose into a globally executable process.

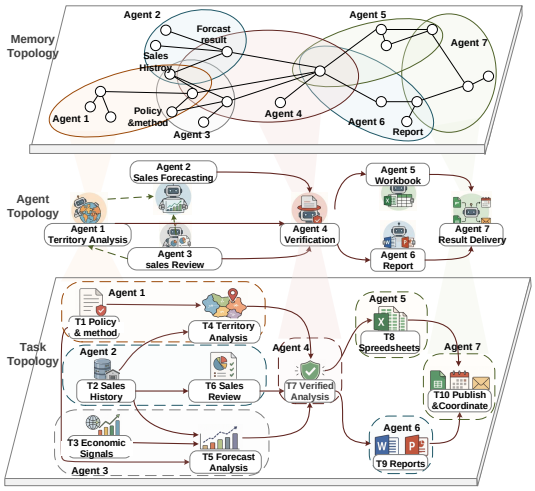



Figure 1: **Topological coherence in ToCoMAS.** Task topology jointly induces agent responsibilities, collaboration interfaces, and memory boundaries, which evolve as a coupled organization.

Recent work automates these choices by searching computation graphs, agentic programs, or workflows (Zhuge et al., 2024; Hu et al., 2025; Zhang et al., 2025c); adapting communication topology or task orchestration (Liu et al., 2024; Zhang et al., 2025b; Yang et al., 2025; Wu et al., 2026; Ke et al., 2026); and retaining structured or decentralized experience (Zhang et al., 2025a; Wang et al., 2026; Yang et al., 2026; Hao et al., 2026). Evolutionary methods further generate



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

agents (Yuan et al., 2025). The most comprehensive system-level formulation jointly evolves agent roles, prompts, models, tools, and communication topology, while retaining a candidate pool and experience memory for cross-task reuse (Hu et al., 2026). Nevertheless, task workflows, agent responsibilities, collaboration, and memory remain separately represented structures, without an explicit constraint that keeps their relations compatible as the system changes.

Figure 1 illustrates this relation on a query whose policy, sales-history, and economic-evidence branches feed workbook, reporting, and delivery tasks. The workflow itself supplies a common organizational reference: task regions delimit responsibility and tool scope, cross-region dependencies create handoffs and optional verification, and ownership boundaries delimit private and selectively shared memory. These are different views of one task-conditioned organization. We call their agreement *topological coherence*: not similarity between graph shapes, but consistency among the relations each layer derives from the task topology (Bodnar et al., 2021).

Building on this formulation, we propose **TOCOMAS**, a *Topology-Coherent Multi-Agent System* that carries this constraint through organization and self-evolution. TOCOMAS maps an upstream-provided task graph to a public tool graph under global operation and dependency constraints, then groups structurally and operationally compatible regions into reusable agent responsibility domains, allowing one agent to own multiple related subgraphs while separating heterogeneous execution semantics. It then derives collaboration and memory coordination from relations among these domains: responsibility ownership delimits local tools and private memory, task-induced adjacency provides dependency-aware handoffs, and boundary policies regulate selective sharing and verifier escalation. Finally, an online controller proposes changes to reusable structural policies and compiles each candidate against the task topology. It reconciles affected ownership, handoff, and memory relations, then retains structurally valid candidates only when their evaluated reward improves on the parent. We evaluate fixed and evolving variants against a direct-call control, prespecified and automatically designed baselines, and a self-evolving baseline across reasoning, tool-use, and software-engineering benchmarks. Our contributions are:

- We formulate *topological coherence* as a structural constraint that keeps task workflows, agent responsibilities, collaboration interfaces, and memory boundaries consistent during self-evolution.
- We develop a topology-grounded architecture in which task and tool relations induce reusable agent responsibility domains, dependency-aware and profile-conditioned collaboration, and boundary-regulated memory access; execution feedback guides coupled structural proposals.
- We establish controlled comparisons with prespecified, automatically designed, and self-evolving MAS baselines to separate architectural gains from evolutionary search.

## 2 RELATED WORK

Early role-based and conversational MAS organize collaboration through role play, proposal aggregation, debate, and peer review (Li et al., 2023; Wu et al., 2024; Chen et al., 2024; Du et al., 2024; Xu et al., 2023); MetaGPT and ChatDev encode specialized roles in manually specified software-development workflows (Hong et al., 2024; Qian et al., 2024). Automatic design searches optimizable computation graphs (GPTSwarm) (Zhuge et al., 2024), code-defined agentic systems and workflow programs (ADAS and AFlow) (Hu et al., 2025; Zhang et al., 2025c), or agents, prompts, and communication structures (AutoAgents, MASS, and ARG-Designer) (Chen et al., 2023; Zhou et al., 2026; Li et al., 2026b). MAS-Orchestra adds holistic task-conditioned orchestration (Ke et al., 2026). These methods move beyond fixed protocols but favor atomic roles or workflow nodes over reusable task-and-tool responsibility domains.

Dynamic agent networks adapt participating agents at inference time (Liu et al., 2024); G-Designer learns task-aware communication graphs, AgentNet evolves decentralized coordination, and CARD conditions communication topology on dynamic environments (Zhang et al., 2025b; Yang et al., 2025; Wu et al., 2026). Graph-conditioned language models and graph retrieval guide reasoning and information selection (Tang et al., 2024; He et al., 2024), while topological learning studies local-to-global compatibility through shared incidence and boundaries (Hansen & Ghrist, 2019; Bodnar et al., 2021). In memory organization, G-Memory stores hierarchical collaboration experience, E-mem reconstructs episodic context through local assistants, and PlugMem organizes reusable knowledge in a typed graph (Zhang et al., 2025a; Wang et al., 2026; Yang et al., 2026). Together, these lines

support relational reasoning, communication, and memory; they do not, by themselves, jointly derive task ownership, tool authority, required handoffs, and memory visibility from a shared task-and-tool topology.

Recent self-evolution methods revise organizations using feedback: EvoAgent expands agent populations through mutation and crossover, while DecentMem uses decentralized private memories for agent-level self-evolution (Yuan et al., 2025; Hao et al., 2026). EvoMAS, the closest system-level approach, represents a MAS as a structured configuration and jointly evolves agent roles, prompts, model and tool assignments, and communication topology; it also retains a candidate pool and experience memory for cross-task reuse (Hu et al., 2026). Yet co-editing these components does not itself require changes in responsibility to induce compatible agent capabilities, handoffs, and memory permissions after a structural change. TOCOMAS instead enforces *topological coherence*, deriving agent responsibilities, required handoffs, and memory boundaries from shared task-and-tool regions.

## 3 PROBLEM FORMULATION

Following a configuration-based formulation for evolutionary multi-agent systems (Hu et al., 2026), a multi-agent system is specified by:

$$C = (G, \mathcal{A}, \mathcal{V}_{\text{in}}, \mathcal{V}_{\text{out}}), \tag{1}$$

where $G = (\mathcal{V}, \mathcal{E})$ is a directed communication graph over agents, $\mathcal{A} = \{A_i\}_{i=1}^{k}$ is the corresponding set of agent configurations, and $\mathcal{V}_{\text{in}}, \mathcal{V}_{\text{out}} \subseteq \mathcal{V}$ denote the input and output agents. Each $A_i = (b_i, p_i, \Gamma_i)$ contains a backbone model, prompt, and accessible tool set. Given a task $q \in \mathcal{Q}$ with task graph $T_q = (\mathcal{U}_q, \mathcal{D}_q)$, let $H_q = (\mathcal{U}_q, \mathcal{D}_q, \Psi_q)$ be its tool-grounded executable topology. The responsibility map $\pi_q : \mathcal{U}_q \to \mathcal{S}_q$ groups task nodes into regions, and the ownership map $\phi_q : \mathcal{S}_q \to \mathcal{V}$ assigns these regions to capable agents. The task-conditioned configuration is:

$$C_q = (C, H_q, \pi_q, \phi_q, \Lambda_q, \mathcal{M}_q), \qquad \overline{H}_q = H_q/\pi_q. \tag{2}$$

Here $C$ is instantiated with the task-conditioned collaboration graph $G_q$. The policy $\Lambda_q$ governs memory admission and visibility, whereas $\mathcal{M}_q$ denotes runtime memory records. We call $C_q$ topology-coherent when responsibility regions have capable owners, required cross-region dependencies have compatible handoffs, and memory visibility respects the same ownership boundaries. Executing $C_q$ on $q$ yields an output $y_q$ and trace $\tau_q$; configuration quality is evaluated by:

$$\begin{aligned} (y_q, \tau_q) &= \text{Exec}(C_q, q), \\ R(q, C_q) &= \text{Metrics}(q, C_q) - \beta \cdot \text{Cost}(C_q). \end{aligned} \tag{3}$$

Here $\text{Metrics}(q, C_q)$ measures accepted execution quality and may incorporate bounded evidence of useful memory context, while $\text{Cost}(C_q)$ denotes the measured resource cost used in candidate selection. The scalar $\beta \geq 0$ controls the performance–efficiency trade-off. The trace provides feedback for structural proposals; candidates are retained only when they pass executable structural checks and improve accepted reward. Accumulated traces inform subsequent evolution and selection.

## 4 THE PROPOSED MODEL

We present **TOCOMAS**, a topology-coherent framework for constructing and evolving multi-agent systems. Its central principle is that agent organization, collaboration, and memory should not be designed as independent configuration choices. Instead, they are derived from a common task topology and jointly screened as the system evolves. Given a task graph $T_q$ and the public tool graph $K$, TOCOMAS first grounds every existing task node by a whole-graph map $\Psi_q$, without redecomposing the user query. It then forms a quotient topology over responsibility regions, assigns these regions to capable agents, and derives communication and memory boundaries from the same maps. This construction is summarized by:

$$T_q \xrightarrow{\Psi_q} H_q \xrightarrow{\pi_q} \overline{H}_q, \qquad \phi_q : \mathcal{S}_q \longrightarrow \mathcal{V}_q. \tag{4}$$

Here $\pi_q$ groups task nodes into responsibility regions, $\overline{H}_q$ preserves their cross-region dependencies, and $\phi_q$ assigns regions to agents. These maps determine required handoffs and constrain the collaboration graph $G_q$ and memory policy $\Lambda_q$; additional communication and retrieved records remain subject to their respective policies. As illustrated in Figure 2, the four modules below realize this construction and preserve it during structural evolution.

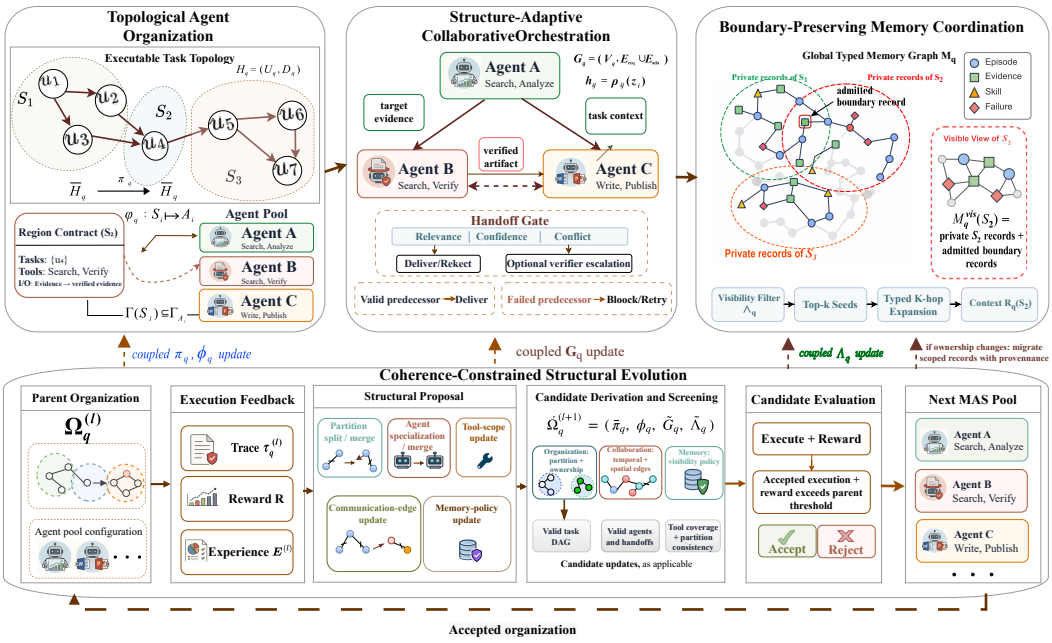



Figure 2: **Overview of ToCoMAS.** Task-derived responsibility regions organize agents, handoffs, and memory boundaries; online evolution preserves their coherence.

### 4.1 Topological Agent Organization

**Task topology and responsibility quotient.** For a task $q$, let $T_q = (\mathcal{U}_q, \mathcal{D}_q)$ be the given task graph and let $K = (\mathcal{T}, \mathcal{R})$ be a public graph of tool interfaces and their operated resources. Each node exposes an operation requirement $R(u)$—including operations, resources, effects, receipt types, and declared input–output semantics. The grounding map $\Psi_q : T_q \to 2^{\mathcal{T}}$ selects a minimal valid tool bundle $\Gamma(u) = \Psi_q(u)$ for every existing node jointly, subject to capability coverage, edge-semantic compatibility, side-effect constraints, and provider continuity. Gold trajectories and evaluator state are excluded from $\Psi_q$. The grounded executable topology is $H_q = (\mathcal{U}_q, \mathcal{D}_q, \Psi_q)$. Rather than instantiating one agent per subtask, ToCoMAS groups operationally compatible nodes into responsibility regions. We express this organization by a surjective map:

$$\pi_q : \mathcal{U}_q \longrightarrow \mathcal{S}_q, \qquad \mathcal{U}_q = \coprod_{S \in \mathcal{S}_q} S, \quad \pi_q(u) = S \text{ for } u \in S, \tag{5}$$

where each $S \in \mathcal{S}_q$ is a responsibility region and the disjoint union assigns every task node to exactly one such region. Contracting the regions under $\pi_q$ yields the quotient topology:

$$\overline{H}_q = H_q/\pi_q = \left(\mathcal{S}_q, \overline{\mathcal{D}}_q\right), \qquad (S_i, S_j) \in \overline{\mathcal{D}}_q \iff \exists (u, v) \in \mathcal{D}_q : \pi_q(u) = S_i,\ \pi_q(v) = S_j, \tag{6}$$

for $S_i \neq S_j$. Thus, $\overline{H}_q$ preserves dependencies between regions while hiding dependencies internal to a single responsibility domain. To preserve executability, a region merge is admissible only when the resulting quotient remains acyclic. To construct $\pi_q$, we associate each pair of task nodes with the compatibility profile:

$$\mathbf{a}_q(u, v) = \left(a_\Gamma(u, v), a_\mathcal{Y}(u, v), a_\mathcal{D}(u, v), a_\mathbb{S}(u, v)\right), \tag{7}$$

whose components respectively measure tool overlap, input–output compatibility, dependency adjacency, and semantic similarity. Starting from singleton regions, operationally compatible regions are merged only when the resulting region respects the prescribed task and tool capacities and the quotient remains acyclic. Thus, $\pi_q$ uses complementary structural evidence without introducing a separate free-form role decomposition.

**Capability-grounded ownership.** Let $\Gamma(S) = \bigcup_{u \in S} \Gamma(u)$ be the capabilities required by a region. A responsibility assignment is a map:

$$\phi_q : \mathcal{S}_q \longrightarrow \mathcal{V}_q, \qquad \Gamma(S) \subseteq \Gamma_{\phi_q(S)} \quad \text{for every } S \in \mathcal{S}_q, \tag{8}$$

where $\mathcal{V}_q$ is the active agent set and $\Gamma_A$ denotes the tools available to agent $A$. The map need not be injective: one agent may own several compatible regions, while structurally or operationally distinct regions remain assigned to different agents. If no existing agent covers a region, the agent pool supplies a new specialization whose capabilities cover $\Gamma(S)$. In this way, an agent role is defined by a responsibility domain in the task topology rather than by a prefabricated textual label.

### 4.2 Structure-Adaptive Collaborative Orchestration

The quotient topology determines which responsibility domains must exchange information. Pushing cross-region task dependencies through the ownership map gives the communication edges required for task execution:

$$\mathcal{E}_q^{\mathrm{req}} = \left\{ \big(\phi_q(S_i), \phi_q(S_j)\big) \;\middle|\; \exists (u,v) \in \mathcal{D}_q,\ u \in S_i,\ v \in S_j,\ \phi_q(S_i) \neq \phi_q(S_j) \right\}. \tag{9}$$

The runtime collaboration topology combines these dependency-induced edges with profile-conditioned spatial edges:

$$G_q = \big(\mathcal{V}_q, \mathcal{E}_q^{\mathrm{req}} \cup \mathcal{E}_q^{\mathrm{sp}}\big). \tag{10}$$

Here $\mathcal{E}_q^{\mathrm{sp}}$ is constructed from agent-profile similarity, query relevance, and tool input–output complementarity, providing additional communication opportunities that may be pruned adaptively.

Let $z_i$ denote the artifact produced for region $S_i$. Local execution and boundary-aware handoff are summarized by:

$$\begin{aligned} z_i &= A_{\phi_q(S_i)}\big(q|_{S_i}, \Gamma(S_i), \mathcal{M}_q^{\mathrm{vis}}(S_i), \{h_{ji}\}\big), \\ h_{ij} &= \mathrm{Route}_{ij}(z_i; \xi_{ij}) \in \{\rho_{ij}(z_i), \varnothing\}. \end{aligned} \tag{11}$$

Here $\rho_{ij}$ exposes only the artifact required by the downstream region, whereas $\mathrm{Route}_{ij}$ applies the dynamic topology and routing policy; a verifier request may be triggered when required.

### 4.3 Boundary-Preserving Memory Coordination

ToCoMAS associates each memory record with its producing agent, task region, and provenance. The responsibility topology induces a visibility policy $\Lambda_q$ that determines which private records and admitted boundary evidence are available to a region:

$$\mathcal{M}_q^{\mathrm{vis}}(S_i) = \mathrm{Filter}_{\Lambda_q}\left(\mathcal{M}_q; S_i\right). \tag{12}$$

The filter first enforces ownership and boundary permissions; semantic retrieval subsequently ranks only the admissible records. Specifically, semantic retrieval selects seeds from the admissible view, after which typed graph expansion reconstructs the regional context:

$$\mathcal{R}_q(S_i) = \mathrm{Expand}_L\big(\mathrm{Seed}_k\big(q|_{S_i}, \mathcal{M}_q^{\mathrm{vis}}(S_i)\big); \mathcal{M}_q^{\mathrm{vis}}(S_i)\big), \tag{13}$$

where $k$ is the number of retrieval seeds and $L$ is the expansion depth. Semantic relevance can therefore rank admissible memories but cannot override their ownership or sharing boundary. Records may encode episodic, evidence, skill, subgraph, and failure relations.

The association between experience and responsibility is preserved when agent ownership changes. If region $S$ is reassigned, its private memory is transported with the region:

$$\phi_q^{(t)}(S) \neq \phi_q^{(t+1)}(S) \quad \Longrightarrow \quad \mu_S^{t\to t+1} : \mathcal{M}_q^{\circ,(t)}(S) \longrightarrow \mathcal{M}_q^{\circ,(t+1)}(S), \tag{14}$$

where $\mu_S^{t\to t+1}$ retains provenance while updating access authority. When task-region ownership changes, records supported by the reassigned tasks are transferred with provenance and their access metadata is updated. This prevents both orphaned experience and unintended global sharing.

### 4.4 Coherence-Constrained Structural Evolution

The structural projection of the task-conditioned configuration $C_q$ is:

$$\Omega_q = \mathrm{Struct}(C_q) = (\pi_q, \phi_q, G_q, \Lambda_q). \tag{15}$$

Here $\Lambda_q$ is the memory admission, visibility, retrieval, and transport policy induced by the responsibility quotient; concrete records $\mathcal{M}_q$ are runtime state governed by $\Lambda_q$, not genes in the evolutionary

configuration. Under the coherence requirement above, $\Omega_q$ is coherent when its ownership, collaboration, and memory policy are supported by the quotient topology $\overline{H}_q$. The construction in the preceding modules establishes this relation through the shared maps $\pi_q$ and $\phi_q$, rather than through independent structural penalties.

At evolution step $t$, execution feedback $\tau_q^{(t)}$ and accumulated experience $\mathcal{E}^{(t)}$ guide a structural proposal:

$$\widetilde{\Omega}_q^{(t+1)} = \mathrm{Propose}\Big(\Omega_q^{(t)}, \tau_q^{(t)}, \mathcal{E}^{(t)}\Big). \tag{16}$$

Equation 16 summarizes proposal at the configuration level: evolution changes reusable structural policies, which are compiled into a candidate organization for the current task. A proposal may change only a subset of policies. If it changes task-region ownership, compilation reconciles required handoffs and memory permissions with the new boundaries; other proposals must also preserve those task dependencies. Coupling therefore requires compatible relations, not a change to every component on every step.

A single structural screen checks that each task node has one capable owner and that required cross-region handoffs and memory visibility respect the induced boundaries. A valid candidate is then executed; $\mathrm{Accept}$ denotes reward-based selection against its parent using the prescribed margin:

$$\Omega_q^{(t+1)} = \begin{cases} \widetilde{\Omega}_q^{(t+1)}, & \mathrm{Valid}(\widetilde{\Omega}_q^{(t+1)}) \wedge \mathrm{Accept}(\widetilde{\Omega}_q^{(t+1)}), \\ \Omega_q^{(t)}, & \text{otherwise.} \end{cases} \tag{17}$$

An accepted organization is available to subsequently launched tasks, while execution traces inform later proposals.

# 5 Experiments

We evaluate whether topological coherence across task, agent, collaboration, and memory structures improves task performance, verified workflow progress, handoff reliability, and information isolation. We then disentangle the contributions of the structural components and evolution, and examine parameter sensitivity and token cost conditioned on achieved workflow progress.

## 5.1 Experimental Setup

**Benchmarks.** We evaluate reasoning on **BBEH** (Kazemi et al., 2025), stateful tool use on **WorkBench** (Styles et al., 2024), and repository repair on **SWE-Bench-Verified** (Jimenez et al., 2023), following the general benchmark selection of evolutionary MAS work (Hu et al., 2026). We also use **CoMemBench** (Zhao et al., 2026) to assess multi-agent workflows with executable dependency graphs, typed handoffs, and paired clean and polluted instances. Its native evaluators separate terminal success from verified progress, information transfer, and isolation. Dataset details appear in Appendix A.

**Evaluation Metrics.** We report BBEH accuracy, WorkBench task success, and the official SWE-Bench-Verified resolved rate. For CoMemBench (Zhao et al., 2026), SR measures terminal success, VNCR verified task-node progress, VHS accepted mandatory handoffs, and ICS memory isolation under paired pollution, conditional on clean-target success. Definitions and denominators appear in Appendix A.2. Token cost includes execution, coordination, memory, and recovery calls; infrastructure failures and incomplete telemetry are reported separately.

**Backbone Models.** The main comparison uses **Qwen3.8-27B** and **DeepSeek-V4-Flash**; the separate large-model comparison uses **GPT-6-luna**. Each run uses a homogeneous task-agent backbone, with model, tool schema, context budget, and decoding policy controlled within each comparison.

**Baselines.** We compare **Direct LLM Call** as a control; fixed organizations **Peer Review** (Xu et al., 2023) and **SMoA** (Li et al., 2025); automatic design methods **ADAS** (Hu et al., 2025) and **OMAC** (Li et al., 2026a); and the self-evolving baseline **EvoMAS** (Hu et al., 2026). Benchmark adapters record execution and handoff evidence without changing a method's routing or memory decisions.

**Implementation Details.** Linux clients access the backbones through OpenAI-compatible APIs; Qwen3.8-27B inference runs on A100 GPUs, while CoMemBench execution uses H800 and A100

Table 1: Overall comparison on BBEH, WorkBench, SWE-Bench-Verified, and CoMemBench.

| Backbone | Organization | Method | General Task Performance | | | CoMemBench | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | BBEH ↑ | WorkBench ↑ | SWE-V ↑ | SR ↑ | VNCR ↑ | VHS ↑ | ICS ↑ |
| Qwen3.8-27B | Control | Direct LLM Call | 19.13 | 19.28 | 9.80 | 0.00 | 8.81 | 34.50 | 0.00 |
| | Prespecified | Peer Review | 46.52 | 37.10 | 30.50 | 1.00 | 30.74 | 33.37 | 19.10 |
| | | SMoA | 37.17 | 37.25 | 35.00 | 4.00 | 38.73 | 43.32 | 10.10 |
| | Automatically designed | ADAS | 32.61 | 62.17 | 34.20 | 0.00 | 13.91 | 27.73 | 7.40 |
| | | OMAC | 41.09 | 55.94 | 39.50 | 2.00 | 32.98 | 40.51 | 17.30 |
| | Self-evolving | EvoMAS | 37.60 | 59.13 | 40.00 | 1.00 | 30.42 | 30.73 | 26.80 |
| | Ours | ToCoMAS | 54.78 | 77.68 | 48.00 | 14.30 | 41.60 | 51.00 | 84.90 |
| DeepSeek-V4-Flash | Control | Direct LLM Call | 14.13 | 20.43 | 12.00 | 0.0 | 9.03 | 27.54 | 6.50 |
| | Prespecified | Peer Review | 49.35 | 39.57 | 39.60 | 0.50 | 27.51 | 29.45 | 17.10 |
| | | SMoA | 43.91 | 44.06 | 43.00 | 1.00 | 37.92 | 41.96 | 14.90 |
| | Automatically designed | ADAS | 30.65 | 52.03 | 34.20 | 3.00 | 33.84 | 36.17 | 14.30 |
| | | OMAC | 41.30 | 35.36 | 43.00 | 3.50 | 56.88 | 55.61 | 4.40 |
| | Self-evolving | EvoMAS | 50.65 | 55.50 | 46.00 | 1.50 | 27.99 | 27.17 | 9.80 |
| | Ours | ToCoMAS | 55.20 | 61.45 | 49.00 | 14.00 | 44.36 | 59.92 | 76.50 |

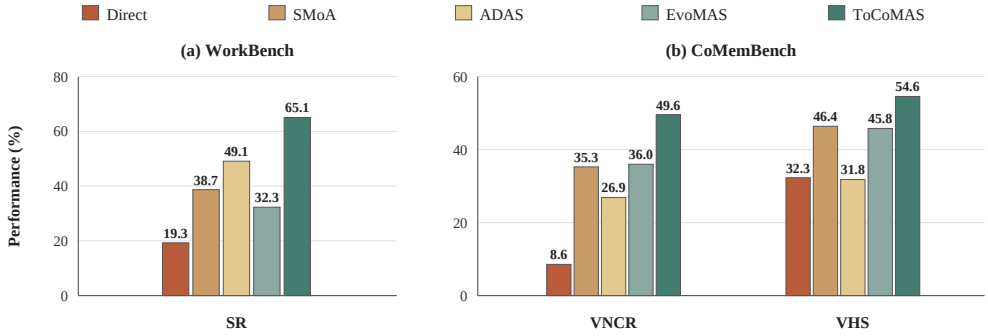



Figure 3: GPT-6-luna results on WorkBench SR and CoMemBench VNCR/VHS.

workers. Methods share benchmark tasks, tools, and native evaluators; paired CoMemBench variants start from the same source state in isolated workspaces. The fixed ToCoMAS configuration uses organization affinity $0.45$ and memory retrieval $k = 3$; we vary these over $\{0.30, 0.45, 0.60, 0.75\}$ and $\{1, 3, 5, 8\}$, respectively. In online structural evolution, a task's first attempt uses the genome current at launch and supplies its reported score; feedback then proposes one structural candidate, tested on that same task. Accepted changes affect subsequently launched tasks, not the scored attempt. We retain native evaluator receipts, handoff traces, and model-call ledgers for auditing and cost accounting.

### 5.2 Main Results

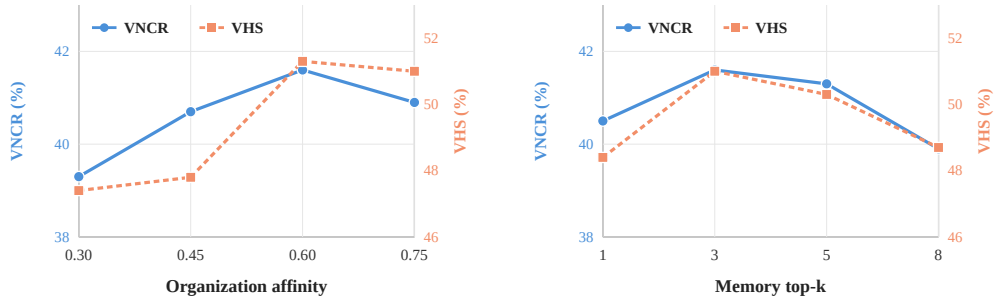
Table 1 jointly reports general task performance and the four CoMemBench quality measures for the Qwen3.8-27B and DeepSeek-V4-Flash backbones. Figure 3 reports the GPT-6-luna comparison. We draw two observations from the reported results:

**Topology-coherent organization improves general task performance.** With Qwen3.8-27B, ToCoMAS reaches 54.78% on BBEH, 77.68% on WorkBench, and 48.00% on SWE-Bench-Verified, exceeding the strongest reported baseline by 8.26, 15.51, and 8.00 percentage points, respectively. With DeepSeek-V4-Flash, the corresponding scores are 55.20%, 61.45%, and 49.00%, also leading the reported baselines. These gains are consistent with grounding agent responsibility and tool access in task regions and routing execution across their dependency boundaries.

**The gains extend to verified handoffs and information isolation.** On Qwen3.8-27B CoMemBench, ToCoMAS attains 14.30% SR, 41.60% VNCR, 51.00% VHS, and 84.90% ICS, compared with

Table 2: Architectural ablations and fixed-versus-evolving comparison of ToCoMAS.

| Variant | WorkBench | CoMemBench | | | |
|---|---|---|---|---|---|
| | SR ↑ | SR ↑ | VNCR ↑ | VHS ↑ | ICS ↑ |
| w/o Topological Agent Organization | 69.86 | 4.00 | 27.27 | 37.97 | 67.14 |
| w/o Structure-Adaptive Collaborative Orchestration | 67.25 | 0.00 | 28.20 | 10.15 | 66.70 |
| w/o Boundary-Preserving Memory Coordination | 68.55 | 6.50 | 36.30 | 42.07 | 64.94 |
| w/o Coherence-Constrained Structural Evolution | 71.16 | 10.60 | 40.07 | 47.80 | 80.80 |
| ToCoMAS | 77.68 | 14.30 | 41.60 | 51.00 | 84.90 |




(a) Organization affinity.

(b) Memory retrieval budget.

Figure 4: Parameter sensitivity of ToCoMAS.

EvoMAS's 1.00%, 30.42%, 30.73%, and 26.80%. Improvement is visible not only at the terminal outcome but also in verified node progress, accepted cross-agent artifacts, and resistance to polluted context. This pattern is consistent with deriving collaboration interfaces and memory visibility from the same responsibility topology; component contributions are examined in the ablations below. With GPT-6-luna, ToCoMAS reaches 65.07% WorkBench SR and 49.58% CoMemBench VNCR, and also leads EvoMAS on VHS (54.57% versus 45.80%). CoMemBench SR is 9.00% for ToCoMAS and at most 0.50% for the compared baselines.

### 5.3 Ablation Study

Table 2 compares full ToCoMAS with variants that remove one structural mechanism at a time. **Topological Agent Organization** assigns connected, tool-compatible task regions to agents; replacing this assignment with canonical-order groups preserves tool coverage but lowers CoMemBench VNCR by 14.33 points. This suggests that the placement of responsibility matters beyond the availability of tools. **Structure-Adaptive Collaborative Orchestration** derives inter-agent handoffs from cross-region task dependencies and checks whether downstream agents can consume the projected artifacts. Replacing these interfaces with fixed predecessor handoffs reduces VHS by 40.85 points and CoMemBench SR to zero, exposing the cost of a broken consumer boundary even when upstream outputs are verified. **Boundary-Preserving Memory Coordination** restricts retrieval to private records and evidence admitted across responsibility boundaries; making every verified record globally visible reduces ICS by 19.96 points. Thus, record validity alone does not determine whether it belongs in an agent's context. Finally, **Coherence-Constrained Structural Evolution** uses task feedback to update the organization for subsequent tasks; retaining a fixed structure reduces WorkBench SR by 6.52 points. Together, these ablations link the three coherence relations to distinct outcomes: responsibility assignment to verified progress, cross-region handoffs to artifact acceptance, and memory boundaries to isolation; online evolution further improves task success.

### 5.4 Parameter Sensitivity

Figure 4 varies two controls while holding the remaining settings fixed. The organization-affinity threshold governs how readily related task nodes form an agent region. Both VNCR and VHS improve from 0.30 to 0.60, where they reach 41.6% and 51.3%, before declining at 0.75. Too little selectivity can mix weakly related responsibilities, while too much can fragment useful regions. The

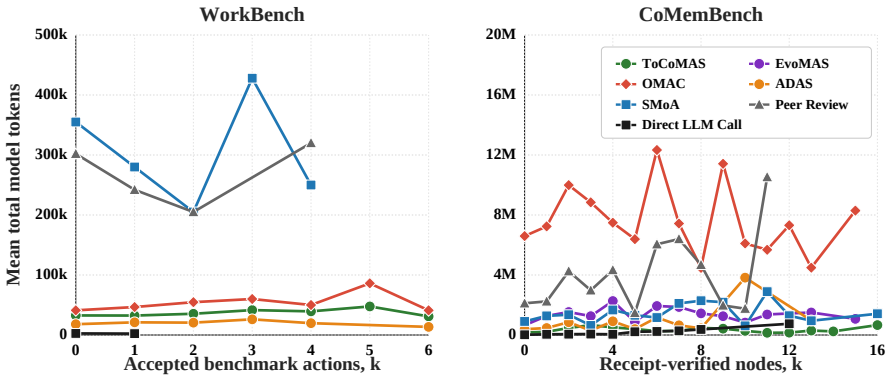



Figure 5: Progress-conditioned token cost across WorkBench and CoMemBench.

memory retrieval budget $k$ limits the records supplied to a task node. Both measures peak at $k = 3$ and decline as retrieval expands to $k = 8$, consistent with extra context offering no guaranteed benefit once relevant evidence is available. Together, the sweeps favor a bounded task-region and memory scope rather than maximal grouping or retrieval.

### 5.5 Cost Analysis

We examine token use within bins of observed progress rather than averaging over tasks of unequal length. For task $q$, let $n(q)$ be the number of evaluator-matched actions on WorkBench or receipt-verified nodes on CoMemBench, and let $c(q)$ be its recorded input and output model tokens. Figure 5 plots the mean $c(q)$ within each observed $n(q) = k$ bin, using successful WorkBench tasks and valid CoMemBench cells with complete token telemetry. The WorkBench ToCoMAS curve pairs post-recheck progress with first-pass tokens; the CoMemBench panel uses a separate cohort.

On WorkBench, the plotted ToCoMAS curve lies below Peer Review and SMoA but above Direct LLM Call and ADAS. On CoMemBench, it lies below the other multi-agent curves at most shared progress levels, although Direct LLM Call remains cheaper. This pattern is consistent with the organization in Figure 1: grouping compatible task nodes internalizes their dependencies within an agent, so required handoffs arise only at cross-region edges; projected artifacts and boundary-filtered retrieval limit the context passed to downstream agents. Aligning these decisions with the task topology can therefore reduce coordination overhead without suppressing verified progress.

## 6 Conclusion

In this work, we presented ToCoMAS, a topology-coherent architecture for self-evolving multi-agent systems. Rather than specifying agent roles, communication, and memory independently, ToCoMAS grounds task nodes in tool requirements, groups compatible regions into agent responsibilities, and derives handoffs and memory boundaries from the resulting organization. Online evolution compiles structural proposals into task-conditioned organizations and retains improvements that satisfy executable and coherence constraints. Experiments on BBEH, WorkBench, and SWE-Bench-Verified show improved task performance over the evaluated baselines; CoMemBench further reveals gains over the self-evolving baseline in verified node completion, accepted handoffs, and isolation from polluted context. Ablations identify distinct contributions from agent organization, collaborative orchestration, memory coordination, and structural evolution. These results support topological coherence as a guiding principle for organizing and evolving multi-agent systems.

### Reproducibility Statement

Experimental settings are detailed in Section 5.1. Native evaluators and retained execution receipts, handoff traces, and model-call ledgers support auditing; the code link appears in the abstract.

## AI USE STATEMENT

Generative AI tools assisted with manuscript drafting and language editing. The authors reviewed AI-assisted material and remain responsible for the scientific claims and reported results.

## REFERENCES

Cristian Bodnar, Fabrizio Frasca, Yuguang Wang, Nina Otter, Guido F. Montufar, Pietro Lio, and Michael Bronstein. Weisfeiler and lehman go topological: Message passing simplicial networks. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 1026–1037, 2021.

Guangyao Chen, Siwei Dong, Yu Shu, Ge Zhang, Jaward Sesay, Börje F. Karlsson, Jie Fu, and Yemin Shi. AutoAgents: A framework for automatic agent generation. *arXiv preprint arXiv:2309.17288*, 2023.

Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chi-Min Chan, Heyang Yu, Yaxi Lu, Yi-Hsin Hung, Chen Qian, et al. AgentVerse: Facilitating multi-agent collaboration and exploring emergent behaviors. In *International Conference on Learning Representations*, 2024.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 11733–11763, 2024.

Taicheng Guo, Xiuying Chen, Yaqi Wang, Ruidi Chang, Shichao Pei, Nitesh V. Chawla, Olaf Wiest, and Xiangliang Zhang. Large language model based multi-agents: A survey of progress and challenges. *arXiv preprint arXiv:2402.01680*, 2024.

Jakob Hansen and Robert Ghrist. Toward a spectral theory of cellular sheaves. *Journal of Applied and Computational Topology*, 3:315–358, 2019.

Guangya Hao, Yunbo Long, and Zhuokai Zhao. Self-evolving multi-agent systems via decentralized memory. *arXiv preprint arXiv:2605.22721*, 2026.

Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh V. Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. G-Retriever: Retrieval-augmented generation for textual graph understanding and question answering. In *Advances in Neural Information Processing Systems*, volume 37, pp. 132876–132907, 2024.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, et al. MetaGPT: Meta programming for a multi-agent collaborative framework. In *International Conference on Learning Representations*, 2024.

Shengran Hu, Cong Lu, and Jeff Clune. Automated design of agentic systems. In *International Conference on Learning Representations*, 2025.

Yuntong Hu, Yuting Zhang, Matthew Trager, Yi Zhang, Shuo Yang, Wei Xia, and Stefano Soatto. EvoMAS: Evolutionary generation of multi-agent systems. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. SWE-bench: Can language models resolve real-world github issues? *arXiv preprint arXiv:2310.06770*, 2023.

Mehran Kazemi, Bahare Fatemi, Hritik Bansal, John Palowitch, Chrysovalantis Anastasiou, Sanket Vaibhav Mehta, Lalit K. Jain, Virginia Aglietti, Disha Jindal, Peter Chen, Nishanth Dikkala, Gladys Tyen, Xin Liu, Uri Shalit, Silvia Chiappa, Kate Olszewska, Yi Tay, Vinh Q. Tran, Quoc V. Le, and Orhan Firat. BIG-Bench Extra Hard. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 26473–26501, 2025.

Zixuan Ke, Yifei Ming, Austin Xu, Ryan Chin, Xuan-Phi Nguyen, Prathyusha Jwalapuram, Semih Yavuz, Caiming Xiong, and Shafiq Joty. MAS-Orchestra: Understanding and improving multi-agent reasoning through holistic orchestration and controlled benchmarks. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

Dapeng Li, Zhen Tan, Peng Qian, Yuheng Li, Kunal Chaudhary, Li Hu, and Jianwei Shen. SMoA: Improving multi-agent large language models with sparse mixture-of-agents. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 54–65. Springer, 2025.

Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. CAMEL: Communicative agents for "mind" exploration of large scale language model society. In *Advances in Neural Information Processing Systems*, volume 36, pp. 51991–52008, 2023.

Shijun Li, Hilaf Hasson, and Joydeep Ghosh. OMAC: A holistic optimization framework for LLM-based multi-agent collaboration. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026a.

Shiyuan Li, Yixin Liu, Qingsong Wen, Chengqi Zhang, and Shirui Pan. Assemble your crew: Automatic multi-agent communication topology design via autoregressive graph generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 23142–23150, 2026b.

Xin Li, Shiyi Wang, Siqi Zeng, Yu Wu, and Yi Yang. A survey on llm-based multi-agent systems: Workflow, infrastructure, and challenges. *Vicinagearth*, 1(1):9, 2024.

Zijun Liu, Yanzhe Zhang, Peng Li, Yang Liu, and Diyi Yang. A dynamic llm-powered agent network for task-oriented agent collaboration. In *First Conference on Language Modeling*, 2024.

Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, et al. ChatDev: Communicative agents for software development. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15174–15186, 2024.

Oliver Styles, Sam Miller, Paloma Cerda-Mardini, Tirthankar Guha, Victor Sanchez, and Bertie Vidgen. WorkBench: A benchmark dataset for agents in a realistic workplace setting. *arXiv preprint arXiv:2405.00823*, 2024.

Jiabin Tang, Yuhao Yang, Wei Wei, Lei Shi, Lixin Su, Suqi Cheng, Dawei Yin, and Chao Huang. GraphGPT: Graph instruction tuning for large language models. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 491–500, 2024.

Kaixiang Wang, Yidan Lin, Jiong Lou, Zhaojiacheng Zhou, Bunyod Suvonov, and Jie Li. E-mem: Multi-agent based episodic context reconstruction for llm agent memory. *arXiv preprint arXiv:2601.21714*, 2026.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, et al. AutoGen: Enabling next-gen llm applications via multi-agent conversations. In *First Conference on Language Modeling*, 2024.

Tongtong Wu, Yanming Li, Ziye Tang, Chen Jiang, Linhao Luo, Guilin Qi, Shirui Pan, and Gholamreza Haffari. CARD: Towards conditional design of multi-agent topological structures. In *International Conference on Learning Representations*, 2026.

Zhenran Xu, Senbao Shi, Baotian Hu, Jindi Yu, Dongfang Li, Min Zhang, and Yuxiang Wu. Towards reasoning in large language models via multi-agent peer review collaboration. *arXiv preprint arXiv:2311.08152*, 2023.

Ke Yang, Zixi Chen, Xuan He, Jize Jiang, Michel Galley, Chenglong Wang, Jianfeng Gao, Jiawei Han, and ChengXiang Zhai. PlugMem: A task-agnostic plugin memory module for llm agents. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

Yingxuan Yang, Huacan Chai, Shuai Shao, Yuanyi Song, Siyuan Qi, Renting Rui, and Weinan Zhang. AgentNet: Decentralized evolutionary coordination for llm-based multi-agent systems. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Siyu Yuan, Kaitao Song, Jiangjie Chen, Xu Tan, Dongsheng Li, and Deqing Yang. EvoAgent: Towards automatic multi-agent generation via evolutionary algorithms. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 6192–6217, 2025.

Guibin Zhang, Muxin Fu, Kun Wang, Guancheng Wan, Miao Yu, and Shuicheng Yan. G-Memory: Tracing hierarchical memory for multi-agent systems. In *Advances in Neural Information Processing Systems*, volume 38, 2025a.

Guibin Zhang, Yanwei Yue, Xiangguo Sun, Guancheng Wan, Miao Yu, Junfeng Fang, Kun Wang, Tianlong Chen, and Dawei Cheng. G-Designer: Architecting multi-agent communication topologies via graph neural networks. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 76678–76692, 2025b.

Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, Xionghui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, Bingnan Zheng, Bang Liu, Yuyu Luo, and Chenglin Wu. AFlow: Automating agentic workflow generation. In *International Conference on Learning Representations*, 2025c.

Sen Zhao, Ruiqi Kong, Zuyu Zhang, Lifeng Shen, Xinyu He, Ding Zou, Xu Zhang, and Qinghua Zhang. CoMemBench: Benchmarking collaborative memory boundaries across multi-agent workflow topologies. `https://comembench.world`, 2026.

Han Zhou, Xingchen Wan, Ruoxi Sun, Hamid Palangi, Shariq Iqbal, Ivan Vulić, Anna Korhonen, and Sercan Ö. Arık. Multi-agent design: Optimizing agents with better prompts and topologies. In *International Conference on Learning Representations*, 2026.

Mingchen Zhuge, Wenyi Wang, Louis Kirsch, Francesco Faccio, Dmitrii Khizbullin, and Jürgen Schmidhuber. GPTSwarm: Language agents as optimizable graphs. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 62743–62767, 2024.

# A Benchmark and Metric Details

## A.1 Benchmark Instances

**BBEH** (Kazemi et al., 2025) tests difficult reasoning questions against benchmark answers. **WorkBench** (Styles et al., 2024) tests workplace tasks involving application tools. **SWE-Bench-Verified** (Jimenez et al., 2023) tests repository patches against its official resolution criterion. We use each benchmark's native evaluator.

**CoMemBench** (Zhao et al., 2026) provides executable workflows in four domains: stateful tool use, repository code, offline retrieval, and formal mathematics. It contains 800 clean workflows (200 per domain) and 800 matched polluted variants. A workflow specifies a request, canonical task nodes and dependency edges, node-local execution requirements, typed handoff contracts, resources, and a domain-native evaluator. Dependency edges represent required task relations, not predefined agent communication edges. Verified node receipts and artifact-exposure records support the progress and handoff metrics below. In a matched polluted variant, one contextually inapplicable artifact is added to a target consumer's incoming artifact view; the query, graph, initial state, tools, and non-target inputs remain unchanged.

## A.2 Metric Definitions and Aggregation

For BBEH, WorkBench, and SWE-Bench-Verified, we report their respective benchmark-level metrics: answer accuracy, task success rate, and official resolved rate. The definitions below apply specifically to CoMemBench.

For CoMemBench, let $\mathcal{C}$ be the evaluated clean workflows. The canonical required nodes of workflow $q$ are $\mathcal{U}_q$, consistent with $T_q$ in Section 4; let $\mathcal{U}_q^{\mathrm{ver}} \subseteq \mathcal{U}_q$ contain nodes with valid native-verifier receipts. Terminal *Success Rate* (SR) and *Verified Node Completion Rate* (VNCR) are:

$$\mathrm{SR} = \frac{1}{|\mathcal{C}|} \sum_{q \in \mathcal{C}} \mathbb{I}[\mathrm{terminal}(q) = 1], \qquad \mathrm{VNCR} = \frac{1}{|\mathcal{C}|} \sum_{q \in \mathcal{C}} \frac{|\mathcal{U}_q^{\mathrm{ver}}|}{|\mathcal{U}_q|}. \tag{18}$$

VNCR averages within-workflow fractions, so workflows receive equal weight; method-private nodes and repeated attempts at a canonical node do not add to the numerator.

Let $\mathcal{H}$ contain each non-root consumer whose mandatory direct predecessors have verified. A consumer belongs to $\mathcal{H}^+$ only if all required predecessor artifacts are exposed at the declared boundary, current, receipt-valid, explicitly consumed, and followed by native-verifier success. *Verified Handoff Success* (VHS) measures required sharing:

$$\mathrm{VHS} = \frac{|\mathcal{H}^+|}{|\mathcal{H}|}. \tag{19}$$

For isolation, each matched pair $z$ has a target consumer $t_z$. Let $\mathcal{P}$ contain pairs whose target verifies in the clean run, and let $\mathcal{P}^+$ contain those pairs whose target also verifies after the inapplicable artifact is introduced in the polluted run. *Isolation Challenge Success* (ICS) measures the preservation of a valid target outcome under this memory-boundary intervention:

$$\mathrm{ICS} = \frac{|\mathcal{P}^+|}{|\mathcal{P}|}, \qquad \mathcal{P} = \bigcup_{d \in \mathcal{D}} \mathcal{P}_d, \tag{20}$$

where $\mathcal{D}$ is the set of four CoMemBench domains. The aggregate is micro-pooled over eligible target pairs, not averaged over domain percentages; eligibility requires clean *target* success, not full clean-workflow success. An eligible polluted target that is not reached or fails verification does not enter $\mathcal{P}^+$. ICS does not require an explicit rejection message. For VHS or ICS, a cohort with no eligible cases is reported as unavailable rather than zero; infrastructure failures are tracked separately from method outcomes. All four CoMemBench ratios are multiplied by 100 when reported as percentages.