MemCo: MEMORY-CENTRIC COLLABORATION FOR GENERALIZING LLM AGENTS TO UNSEEN ENVIRONMENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language model (LLM) agents interact with environments through sequential observation, action, and feedback. While memory enables agents to reuse past experience, existing approaches often construct and maintain memory for individual agents independently, making it costly to collect sufficient trajectories and limiting generalization to unseen environments. Sharing experience accumulated by multiple agents interacting with different environments can reduce repeated exploration. In this work, we ask: *how can shared experience be organized into useful memory and effectively used to support another agent's current decision?* We propose MemCo, a memory-centric collaboration framework for sharing and reusing agent experience. MemCo maintains complementary local and global memory spaces, preserving environment-specific experience locally while reusable records abstracted from local trajectories are aggregated across agents and promoted to global memory based on accumulated evidence. During online interaction, MemCo retrieves and composes memories from both levels based on the agent's current state and decision phase, grounding shared experience in the current environment to support the next decision. Extensive studies on ALFWorld, PDDL, FEVER, and ScienceWorld show that MemCo improves task success under evaluation across seen and unseen environments, as well as reduces redundant exploration compared with the state-of-the-art methods. Our code is available at project.

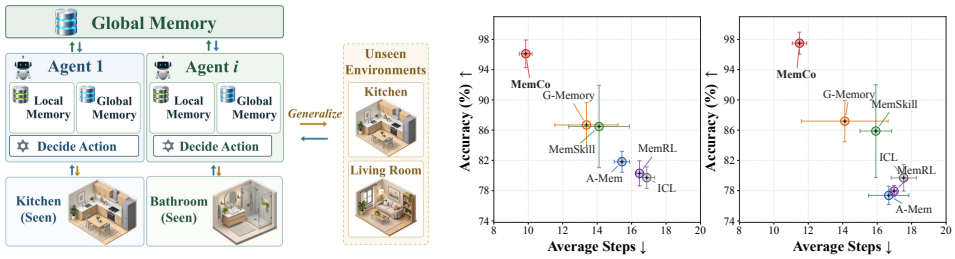
1 INTRODUCTION

Large language model (LLM) agents are increasingly applied to interact with environments, where they make decisions based on the current state and execute actions toward the task goal. Retrieving episodic memories allows agents to reuse interaction experience at inference time (Cao et al., 2025; Fang et al., 2025; Liang et al., 2026). Methods such as Reflexion (Shinn et al., 2023), MemRL (Zhang et al., 2026b), and A-Mem (Xu et al., 2025) improve how individual agents organize and reuse prior experience. However, each agent still needs to collect useful trajectories through its own interactions, while limited experience may provide incomplete or unreliable guidance (Kwon et al., 2025).

When multiple agents operate independently in different environments, experience acquired by one agent provides a natural reference for others. As illustrated in Fig. 1(a), agents can preserve experience from their own environments while sharing reusable experience that may support other agents, including those executing tasks in unseen environments, i.e., environments they have not previously interacted with. This motivates us to ask: *how can one agent's experience help other agents and vice versa?* Existing methods such as G-Memory (Zhang et al., 2025), PlugMem (Yang et al., 2026), and MCMA (Liang et al., 2026) organize or transfer interaction experience across tasks or agents, demonstrating the potential of sharing experience. As shown in Fig. 1(b,c), G-Memory improves over in-context learning (ICL) and agents with isolated memory, yet still leaves room for improving task success and interaction efficiency, particularly on held-out layouts. Yet shared experience is not inherently useful across environments or decision contexts. This raises a crucial challenge: *how can shared experience be organized into useful memory and effectively used to support another agent's current decision?*

To address this challenge, we propose MemCo, a Memory-centric Collaboration framework with complementary local and global memory spaces. As shown in Fig. 2, MemCo contains three main modules. In each agent, *local memory construction* converts interaction trajectories into a local memory graph,

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.



(a) Individual-Agent Interaction (b) Familiar Layout (c) Held-out Layout

Figure 1: (a) Memory-centric collaboration among independently acting agents: each agent preserves environment-specific local experience while contributing reusable experience to global memory for use across environments. (b) and (c) ALFWorld performance on familiar and held-out layouts, respectively, with Qwen3-32B. Markers and error bars denote the mean and standard deviation.

preserving environment-specific context while extracting experience patterns, actionable knowledge, and failure reflections. To form useful shared memory, *progressive global memory collaboration* abstracts selected local records while retaining their applicability conditions, aggregates evidence across agents, and promotes sufficiently supported records to global memory. To translate memory into effective decision guidance, *query-guided memory support* retrieves local and global memories according to the current decision context, filters incompatible guidance, and grounds shared experience in the current environment before composing memory support for the next decision. Thus, `MemCo` enables agents to benefit from experience accumulated by others without blindly transferring source-environment-specific information, while preserving the local grounding required for task execution in unseen environments.

Our main contributions are: (1) We introduce memory-centric collaboration for agents interacting with environments, where experience accumulated across different environments can be shared to reduce repeated exploration and support task execution in unseen environments. (2) We develop `MemCo` with complementary local and global memory, turning shared experience into useful memory through evidence-aware global promotion and into decision guidance related to execution state. (3) Experiments across four benchmarks (ALFWorld, PDDL, FEVER, and ScienceWorld) and multiple LLM backbones show improved task performance with generally fewer interaction steps.

2 Related Work

Individual Experience and Agent Memory. Early agent paradigms such as ReAct (Yao et al., 2022) rely on the current interaction context without persistent experience across trials. Long-term memory systems subsequently support persistent storage and retrieval across interactions (Park et al., 2023; Packer et al., 2023; Zhong et al., 2024; Chhikara et al., 2025; Wang & Chen, 2025; Kwon et al., 2025). For interactive agents, prior work further turns trajectories into reflections, skills, procedural knowledge, or adaptive memory (Shinn et al., 2023; Wang et al., 2023; Zhu et al., 2023; Ding et al., 2026; Fang et al., 2025; Cao et al., 2025; Wu et al., 2025; Xu et al., 2025; Nan et al., 2025; Zhang et al., 2026b;a). These methods mainly improve how experience accumulated within an individual agent or memory system is represented and reused.

Agent Collaboration through Shared Experience. Beyond communication, agent selection, or execution-time orchestration (Wu et al., 2024; Jia et al., 2025), agents can collaborate by reusing experience acquired in prior interactions. G-Memory (Zhang et al., 2025) organizes cross-trial collaboration memory within multi-agent systems, while MATM (Kim et al., 2026) supports population-level sharing of producer-agent trajectories. Agent KB (Tang et al., 2025) enables structured experience reuse across heterogeneous agent frameworks, whereas MemCollab (Chang et al., 2026) and FedAgentKE (Li et al., 2026) construct shared knowledge across different models or agent frameworks. Collectively, these works demonstrate the value of sharing agent experience. For agents interacting with different environments, however, shared experience is not uniformly useful: its value can depend on the receiving environment and current decision. `MemCo` addresses this setting by preserving local grounding while sharing reusable experience across agents.

Constructing and Using Shared Memory. Prior work studies how reusable experience can be represented as transferable skills, rules, workflows, or structured memories (Ding et al., 2026; Fang

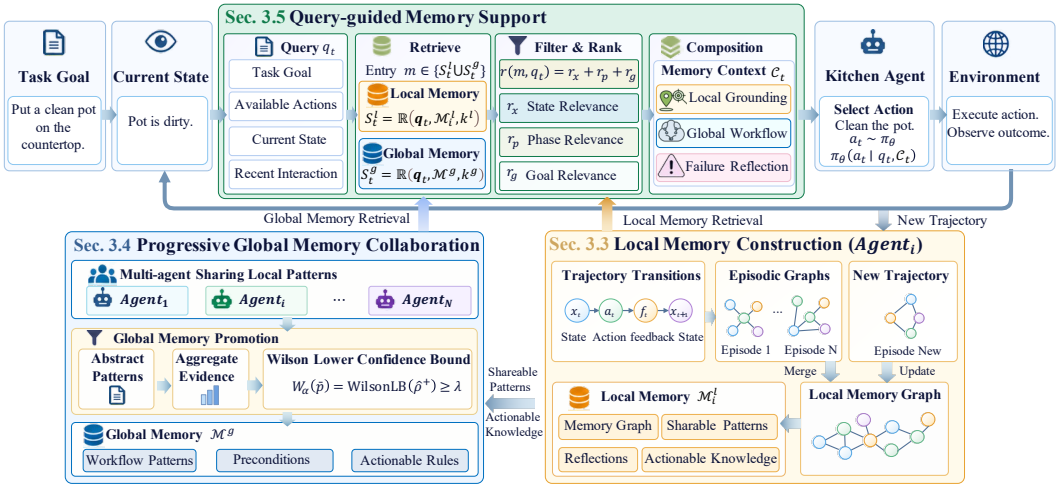


Figure 2: Overview of MemCo. *Local memory construction* organizes environment-specific interaction experience and extracts reusable records. *Progressive global memory collaboration* aggregates evidence across agents and promotes supported reusable records to global memory. *Query-guided memory support* grounds and composes local and global memory for the current decision.

et al., 2025; Cao et al., 2025; Wu et al., 2025; Xiang et al., 2026; Yang et al., 2026; Liang et al., 2026; Xia et al., 2026), as well as how stored memory can be selectively used at inference time (Liu et al., 2025; Zhao et al., 2026). MemCo jointly addresses both stages through evidence-aware global promotion and query-guided local and global memory use, while retaining the local grounding needed for execution. A more comprehensive discussion is provided in Appendix A.

3 Methodology

3.1 Problem Formulation

Among N agents, each agent i independently interacts with an environment using an LLM-based policy model π_θ. At step t, the agent receives a task goal $\boldsymbol{g}$, a state observation $\boldsymbol{x}_t$, and an available action set $\mathcal{A}_t$. It selects an action $\boldsymbol{a}_t \in \mathcal{A}_t$, and the environment returns feedback $\boldsymbol{f}_t$ indicating state progress, pause, failure, or success. An episode trajectory is $\tau = \{(\boldsymbol{x}_t, \boldsymbol{a}_t, \boldsymbol{f}_t, \boldsymbol{x}_{t+1})\}_{t=1}^{T}$. Agent i accumulates experience from environments it has interacted with, but is expected to execute tasks in *unseen environments*, i.e., environments it has not interacted with before evaluation. The next action is selected using context $\mathcal{C}_t$ composed from memory available to the agent:

$$\boldsymbol{a}_t \sim \pi_\theta(\boldsymbol{a}_t \mid \boldsymbol{g}, \boldsymbol{x}_t, \mathcal{A}_t, \mathcal{C}_t). \tag{1}$$

We aim to organize experience accumulated across agents into reusable memory and selectively use it to support independent task execution in seen and unseen environments.

3.2 Framework Overview

As in Fig. 2, MemCo maintains local memory $\mathcal{M}_i^l$ for environment-specific experience and global memory $\mathcal{M}^g$ for reusable experience shared across agents. The framework addresses two stages: constructing memory from accumulated experience and using it for the current decision. For memory construction, *local memory construction* organizes each agent's episodes into a graph and extracts patterns, actionable knowledge, and reflections, while *progressive global memory collaboration* abstracts selected shareable records, aggregates evidence across agents, and promotes supported records to global memory. For decision-time use, *query-guided memory support* selectively retrieves and composes relevant local and global memories into context $\mathcal{C}_t$ for the current decision.

At step t, agent i selects an action according to Eq. 1 and uses environment feedback to form the next query, repeating this process until the episode terminates. During memory acquisition, completed episodes update local memory and provide selected records and evidence for global promotion.

Subsequent queries can access the updated memories, enabling agents to benefit from one another's experience while executing their own tasks independently. The algorithm details are in Appendix B.1.

3.3 Local Memory Construction

Local memory construction organizes an agent's interaction episodes into a graph that connects related state–action transitions and their feedback across episodes. The graph preserves the context in which individual experiences occur and exposes recurring structures across episodes, from which MemCo induces shareable records together with their supporting evidence. This provides environment-specific memory for the agent itself and candidate records for cross-agent sharing.

For each episode τ_n of agent i, MemCo constructs an episodic graph $G_{\tau_n} = (V_{\tau_n}, E_{\tau_n})$ whose nodes v_t^x, v_t^a, and v_t^f store normalized observations, parsed operations, and environment feedback, respectively. Directed edges connect them as temporal transitions: $v_t^x \to v_t^a \to v_t^f \to v_{t+1}^x$. Across N_i episodes of the same agent, nodes are matched when their types and normalized contents agree, and edges are matched when their relations and endpoints agree. Merging these elements gives local memory graph $G_i^l \leftarrow \mathrm{LocalMerge}(\{G_{\tau_n}\}_{n=1}^{N_i})$ as detailed in Algo. 4. The graph and the record sets $\mathcal{P}_i^l$, $\mathcal{E}_i^l$, and $\mathcal{R}_i^l$ derived from the interaction episodes together form the local memory:

$$\mathcal{M}_i^l = \left[G_i^l, \mathcal{P}_i^l, \mathcal{E}_i^l, \mathcal{R}_i^l\right], \tag{2}$$

where $\mathcal{P}_i^l$ contains pattern records of workflows, action preconditions, repairs, and failures, $\mathcal{E}_i^l$ contains actionable-knowledge records, and $\mathcal{R}_i^l$ contains concrete cases and reflection records. We consolidate records of the same type and normalized content, retaining occurrence-level statistics locally and per-episode annotations for promotion.

A local record is shareable when $p \in \mathcal{P}_i^l \cup \mathcal{E}_i^l$. For each such record p, MemCo records supporting, contradicting, and no-progress evidence from its observed occurrences. Before global promotion, annotations associated with the same record, source scope, and episode are deterministically reduced to at most one signed vote. The resulting $n_i^+(p)$ and $n_i^-(p)$ are the episode-reduced supporting and contradicting counts used for promotion, while $n_i^0(p)$ records episode-level stall markers and is kept separate from the signed Wilson sample. Detailed evidence assignment and reduction rules are provided in Appendix B.2.

MemCo updates the graph after each completed episode τ_{new} as $G_i^l \leftarrow \mathrm{LocalMerge}(\{G_i^l, G_{\tau_{\mathrm{new}}}\})$ and merges the newly extracted records and evidence into the existing $\mathcal{P}_i^l$, $\mathcal{E}_i^l$, and $\mathcal{R}_i^l$. Both G_i^l and $\mathcal{R}_i^l$ remain local because they retain environment-specific context. Selected shareable records from $\mathcal{P}_i^l$ and $\mathcal{E}_i^l$, together with their episode-reduced promotion counts, are shared as they become available.

3.4 Progressive Global Memory Collaboration

Progressive global memory collaboration turns selected local experience into reusable memory shared across agents. Each agent abstracts and shares selected local records with their episode-reduced promotion counts, while keeping fine-grained trajectories, local memory graphs, and instance-specific reflections local. MemCo aggregates evidence across agents and promotes supported abstract records into global memory for subsequent queries.

For a shareable local record $p \in \mathcal{P}_i^l \cup \mathcal{E}_i^l$, MemCo constructs its abstract form $\bar{p} = \mathrm{Abstract}(p)$ by replacing concrete entities where supported by the benchmark-specific task-role mapping and removing episode- and scene-specific anchors (Algo. 5). This preserves the record type, applicability conditions, action sequence or relation, and outcome. Records without removable environment-specific anchors can retain their structured form, while records without a transferable global form remain local. For example, a kettle's current location remains local, while a fill-then-heat procedure can be shared using object and tool roles.

Abstract records across agents share the same abstract key when their record types, applicability conditions, action sequences or relations, and outcomes match exactly (Appendix B.3.3). This key determines which local records contribute to the same abstract record. The episode-reduced counts of local records mapping to the same abstract record are aggregated as

$$n^{\omega}(\bar{p}) = \sum_{i \in [N], p' : \mathrm{Abstract}(p') = \bar{p}} n_i^{\omega}(p'), \qquad \omega \in \{+, -\}. \tag{3}$$

Before promotion, fixed type-specific filters exclude incomplete or uninformative abstract records. Promotion further depends on how consistently the aggregated evidence supports each retained record. Let $s(\bar{p}) = n^+(\bar{p}) + n^-(\bar{p})$ denote the signed episode-level evidence size and $\rho^+(\bar{p}) = {}^{n^+(\bar{p})}/_{\max\{s(\bar{p}),1\}}$ its empirical supporting rate for promotion.

However, finite trajectories may provide sparse evidence, so a high empirical supporting rate can arise from only a few episode-level votes. We therefore use the one-sided Wilson lower confidence bound of $\rho^+(\bar{p})$, i.e., $W(\bar{p})$, as a finite-evidence promotion score that accounts for both the empirical supporting rate and evidence size (Wilson, 1927; Brown et al., 2001). Its statistical interpretation and assumptions are discussed in Appendix D.1. Let $z = \Phi^{-1}(1-\alpha)$, where $\alpha \in (0, 1/2)$ is the one-sided tail probability and Φ is the standard normal cumulative distribution function. We set $W(\bar{p}) = 0$ if $s(\bar{p}) = 0$; otherwise,

$$W(\bar{p}) = \frac{s(\bar{p})\rho^+(\bar{p}) + \frac{z^2}{2} - z\sqrt{s(\bar{p})\rho^+(\bar{p})(1-\rho^+(\bar{p})) + \frac{z^2}{4}}}{s(\bar{p}) + z^2}. \tag{4}$$

The following result characterizes the finite-evidence correction.

Theorem 3.1 (Wilson finite-evidence correction)**.** *For $s(\bar{p}) > 0$ and fixed $z > 0$, the correction $\epsilon_W(\bar{p}) = \rho^+(\bar{p}) - W(\bar{p})$ satisfies*

$$0 \le \epsilon_W(\bar{p}) = z\sqrt{\frac{W(\bar{p})(1-W(\bar{p}))}{s(\bar{p})}} \le \frac{z}{2\sqrt{s(\bar{p})}}. \tag{5}$$

Thus, Wilson promotion discounts empirical support under finite evidence, with the maximal correction vanishing uniformly as $O(s(\bar{p})^{-1/2})$.

The proof and exact threshold and sample-size implications are given in Appendix D.

After the fixed type-specific filters above, global memory is initialized as $\mathcal{M}^g = \varnothing$ and updated by merging abstract records whose Wilson score passes threshold λ:

$$\mathcal{M}^g \leftarrow \mathrm{GlobalMerge}\left(\mathcal{M}^g, \{\bar{p} : W(\bar{p}) \ge \lambda\}\right), \quad 0 < \lambda < 1. \tag{6}$$

$\mathrm{GlobalMerge}(\cdot)$ maintains one global record per abstract key, inserting new records or replacing the stored evidence with the latest aggregated counts (Algo. 6). New local records or updated evidence trigger reaggregation and reassessment of the affected abstract records. Existing global records that no longer satisfy the preceding filters or threshold are removed. These updates become available to subsequent queries, allowing each agent to use experience contributed by other agents while executing its own task independently.

3.5 Query-guided Memory Support

To translate stored experience into support for the next decision, `MemCo` combines environment-specific local memory with reusable global memory accumulated across agents. The selected memory must be relevant to the current decision and grounded in the environment the agent is currently interacting with. At step t, the query $\boldsymbol{q}_t = \mathcal{Q}(\boldsymbol{g}, \boldsymbol{x}_t, \mathcal{A}_t, \tau_{<t})$ captures the goal, current observation, available actions, and task progress inferred from the preceding interaction history $\tau_{<t}$. This query guides both memory selection and composition.

We retrieve two sets of memory entries from local and global sources separately:

$$\mathcal{S}_t^l = \mathbb{R}\left(\boldsymbol{q}_t, \mathcal{M}_i^l, k^l\right), \qquad \mathcal{S}_t^g = \mathbb{R}\left(\boldsymbol{q}_t, \mathcal{M}^g, k^g\right), \tag{7}$$

where $\mathbb{R}(\cdot)$ is specified in Algo. 7. The local output $\mathcal{S}_t^l$ is selected from records in $\mathcal{P}_i^l$, $\mathcal{E}_i^l$, and $\mathcal{R}_i^l$, together with bounded transition entries from query-matched states in G_i^l. These entries provide bounded local structural context for assessing workflow or rule applicability. The global output $\mathcal{S}_t^g$ is selected from the promoted records in $\mathcal{M}^g$. Both outputs obey their respective cardinality caps.

Before selection, each memory entry m in the source-specific retrieval pool is scored along three complementary dimensions: state, task phase, and goal:

$$r(m, \boldsymbol{q}_t) = r_x(m, \boldsymbol{q}_t) + r_p(m, \boldsymbol{q}_t) + r_g(m, \boldsymbol{q}_t), \tag{8}$$

where r_x measures consistency with the current state and available actions, r_p measures compatibility with task progress, and r_g measures the alignment of object, destination, or tool roles with the goal.

Selection distinguishes applicability constraints from ranking preferences. Entries are filtered when their required conditions contradict the current state or their actions cannot be matched to $\mathcal{A}_t$ when such matching is required. The remaining entries are ranked using joint relevance, stored evidence, and action compatibility. We reduce priority when stale local information and other soft conflicts occur. Selection retains at most k^l local and k^g global entries. When a retrieval pool exceeds its budget, strong matches along individual relevance dimensions are retained before the remaining positions are filled by rank, preserving complementary support within the limit (Algo. 7).

Composition connects shared knowledge to its local use by grounding global records against $\boldsymbol{x}_t$ and $\mathcal{A}_t$, with local evidence supplying environment-specific context where available. For example, a shared clean-before-place workflow specifies action order, while local evidence identifies a usable basin for cleaning the current object (Appendix B.4.5). The selected entries are organized by their roles in the current decision: *grounding* provides evidence for interpreting the current state, graph relations, and concrete action references; *workflow guidance* provides plans, procedures, and preconditions; and *reflection* provides failure patterns, repairs, and blocked actions. For each support usage role $h \in \mathcal{H} = \{\text{ground}, \text{workflow}, \text{reflection}\}$, let $\mathcal{U}_t^h$ contain the entries assigned to that role. Role-aware composition and query-guided textual rendering produce the final memory context:

$$\{\mathcal{U}_t^h\}_{h\in\mathcal{H}} = \text{Compose}(\mathcal{S}_t^l, \mathcal{S}_t^g \mid \boldsymbol{q}_t, \mathcal{A}_t), \tag{9}$$

$$\mathcal{C}_t = \text{Render}(\boldsymbol{q}_t, \{\mathcal{U}_t^h\}_{h\in\mathcal{H}}) = [c_t^{\text{ground}}, c_t^{\text{workflow}}, c_t^{\text{reflection}}], \tag{10}$$

where c_t^h is the final textual context, functions $\text{Render}(\cdot)$ and $\text{Compose}(\cdot)$ are defined in Algo. 8-9. The resulting $\mathcal{C}_t$ conditions next-action prediction in Eq. 1. After each action, new observations and feedback update the history and the next query, allowing memory support to follow task progress.

4 EXPERIMENTS

Benchmarks. We evaluate MemCo on four benchmarks that necessitate multi-step decision-making across episodes: (1) **ALFWorld** Shridhar et al. (2020) aligns high-level language instructions with low-level robotic actions, challenging agents to reason and act within simulated environments; (2) **PDDL** Ma et al. (2024) evaluates symbolic planning by requiring agents to generate valid action sequences that achieve specified goals under domain constraints; (3) **FEVER** Thorne et al. (2018) evaluates large-scale fact verification, requiring agents to retrieve and reason over external evidence to classify claims as SUPPORTS, REFUTES, or NOT ENOUGH INFO (NEI); and (4) **ScienceWorld** Wang et al. (2022) is a comprehensive interactive benchmark for elementary science experiments.

Baselines. We compare MemCo against several representative baselines: (1) **In-Context Learning** (ICL), using direct in-context prompting without a persistent memory bank; (2) **A-Mem** Xu et al. (2025), a specialized long-term memory architecture; (3) **MemRL** Zhang et al. (2026b), which utilizes reinforcement learning to enable self-evolution via a value-based decision process; (4) **MemSkill** Zhang et al. (2026a), which reframes memory management as a set of learnable natural-language skills; and (5) **G-Memory** Zhang et al. (2025), a hierarchical graph-based memory for multi-agent systems that incorporates interactions and insights.

Implementation Details. We implement MemCo using AutoGen Wu et al. (2024) and evaluate open- and closed-source models. We use $\alpha = 0.05$ and $\lambda = 0.35$ for promotion across all benchmarks. Open-source model inference is conducted on a single NVIDIA RTX PRO 6000 Blackwell GPU. Both local and global retrievers use deterministic bounded selection with facet coverage and rank-based filling, i.e., $k^\ell = k^g = 5$. Unless otherwise specified, we report accuracy (Acc., %) and the average number of interaction steps during evaluation. For each benchmark, memory is constructed from designated source partitions and evaluated on disjoint tasks, with no evaluation trajectories used for memory construction. We use one agent per source domain: four agents for ALFWorld and PDDL, two for FEVER, and ten for ScienceWorld. Each agent is evaluated on the merged test set using its own frozen local memory and the shared global memory, covering both its acquisition domain and other environments. ALFWorld's Familiar and Held-out splits additionally assess layout generalization. Full partitioning and construction details are in Appendix F.

Table 1: Main results on FEVER, PDDL, and ALFWorld (mean ± standard deviation over three runs). **Bold** and underlined indicate the best and second-best results within each backbone, respectively.

Backbone	Method	FEVER		PDDL		ALFWorld			
		Acc. ↑	Steps ↓	Acc. ↑	Steps ↓	Familiar Acc. ↑	Familiar Steps ↓	Held-out Acc. ↑	Held-out Steps ↓
Qwen3-4B	ICL	56.78 ±1.47	4.23 ±0.52	44.44 ±3.47	29.19 ±0.14	60.28 ±4.73	20.43 ±0.18	46.31 ±3.79	22.45 ±0.74
	A-Mem	54.24 ±2.24	4.94 ±1.99	37.22 ±8.42	29.17 ±0.31	58.66 ±8.24	19.88 ±0.93	45.60 ±3.57	22.14 ±0.32
	MemSkill	59.89 ±4.35	3.41 ±0.22	40.97 ±3.49	29.31 ±0.07	61.90 ±6.66	20.00 ±1.07	57.74 ±2.72	20.28 ±1.17
	MemRL	55.93 ±1.70	4.29 ±0.31	46.58 ±5.18	**29.03** ±0.10	59.31 ±4.88	20.60 ±0.22	49.41 ±0.54	22.16 ±0.43
	G-Memory	55.50 ±8.99	5.44 ±4.37	39.38 ±0.88	29.18 ±0.18	62.23 ±2.31	18.73 ±0.61	53.81 ±1.15	20.57 ±1.08
	MemCo	**60.17** ±1.20	**3.38** ±0.48	**53.01** ±1.47	29.16 ±0.21	**84.21** ±2.28	**13.33** ±0.64	**82.84** ±2.65	**13.77** ±0.71
GPT-4o-mini	ICL	63.00 ±3.42	6.33 ±1.10	57.78 ±2.55	28.97 ±0.12	28.57 ±3.08	25.86 ±0.32	35.71 ±2.77	25.53 ±0.74
	A-Mem	61.30 ±4.90	7.02 ±0.87	52.64 ±2.77	29.36 ±0.12	23.05 ±4.82	26.62 ±0.69	35.00 ±2.40	26.04 ±0.75
	MemSkill	62.71 ±3.06	7.76 ±0.21	**59.03** ±4.46	28.89 ±0.17	16.92 ±6.06	27.75 ±1.42	28.93 ±4.43	26.70 ±1.05
	MemRL	62.71 ±2.93	6.57 ±2.11	55.56 ±2.77	28.98 ±0.04	28.24 ±3.27	25.74 ±0.21	34.41 ±0.65	26.10 ±0.30
	G-Memory	62.99 ±6.36	7.95 ±0.26	58.89 ±2.30	**28.41** ±0.19	**42.21** ±2.63	24.05 ±1.23	49.29 ±1.94	24.23 ±1.80
	MemCo	**70.90** ±1.76	**2.63** ±0.18	55.56 ±1.46	29.15 ±0.02	40.26 ±3.10	**22.69** ±0.55	**50.00** ±2.86	**21.75** ±0.61
Qwen3-32B	ICL	66.67 ±1.96	5.66 ±2.24	76.25 ±0.42	26.19 ±0.13	79.72 ±1.42	16.88 ±0.46	79.69 ±1.75	17.54 ±0.73
	A-Mem	68.08 ±1.76	4.40 ±1.93	54.58 ±2.09	25.21 ±0.41	81.82 ±1.39	15.44 ±0.45	77.37 ±1.22	16.67 ±1.17
	MemSkill	67.52 ±3.43	4.92 ±1.63	70.28 ±2.84	26.56 ±0.50	86.48 ±5.45	14.11 ±1.76	85.88 ±6.13	15.92 ±0.92
	MemRL	67.52 ±5.38	5.48 ±1.65	77.78 ±1.05	26.02 ±0.10	80.29 ±1.66	16.46 ±0.20	77.94 ±0.75	16.98 ±0.16
	G-Memory	68.64 ±3.43	6.49 ±1.18	68.47 ±2.10	**24.92** ±0.43	86.69 ±2.94	13.38 ±1.84	87.19 ±2.73	14.12 ±2.52
	MemCo	**71.19** ±2.05	**3.24** ±0.52	**78.89** ±1.58	26.20 ±0.10	**96.62** ±1.82	**9.71** ±0.37	**97.46** ±1.45	**11.23** ±0.42

4.1 Experimental Analysis

Overall Results. Tab. 1 presents the main comparison across three benchmarks and three LLM backbones. For ALFWorld, Familiar and Held-out denote the official `valid_seen` and `valid_unseen` layout splits, respectively. Counting FEVER, PDDL, and ALFWorld for each backbone, MemCo achieves the highest accuracy in 10 out of the 12 evaluation settings and requires the fewest interaction steps in 9 of them. The performance gains are frequently accompanied by shorter trajectories, indicating that MemCo improves task outcomes without relying on additional environment exploration.

The largest gains occur on ALFWorld: with Qwen3-4B, MemCo improves held-out layout accuracy by 43.47% and reduces interaction steps by 32.10% relative to the runner-up baseline (MemSkill). Gains with Qwen3-32B show that benefits extend beyond weaker policy models. Held-out results further support transfer across layouts under this cross-environment evaluation protocol. MemCo also leads FEVER in both accuracy and interaction efficiency under all three backbones. With GPT-4o-mini, it improves accuracy by 12.54% and reduces steps by 58.45% relative to the runner-up baseline. It also improves PDDL accuracy with both Qwen models. These PDDL gains do not always coincide with the fewest interaction steps, showing that success and interaction efficiency are not always aligned. GPT-4o-mini reveals model dependence, but MemCo still achieves the best held-out-layout accuracy and shortest ALFWorld trajectories. Furthermore, exact two-sided Wilcoxon signed-rank tests on the 12 matched configuration means indicate improvements over each baseline in accuracy and steps after Holm correction, under the stated block-level assumptions. Full statistics and effect sizes appear in Appendix H.

Robustness across Additional LLM Backbones. Tab. 2a examines whether the improvements persist across additional LLM families. Across them, MemCo consistently achieves the highest FEVER accuracy, improving over the runner-up baseline by 10.81%, 32.82%, and 6.98%, respectively. These gains generally coincide with shorter trajectories. On DeepSeek-v4-flash, MemCo improves accuracy while requiring nearly the same number of steps as ICL. However, G-Memory did not complete successfully in this setting. Together with Tab. 1, these results indicate that the benefit of MemCo generalizes across model families rather than depending on a particular backbone or model scale.

Generalization to a More Complex Interactive Environment. On ScienceWorld, Tab. 2b shows that MemCo achieves the highest final success and score, and the auxiliary metrics further reveal a distinction between intermediate exploration and successful task completion. G-Memory achieves higher intermediate progress, while MemRL terminates in fewer steps, yet neither converts these advantages into comparable final success. These results show that intermediate progress and fewer steps alone do not ensure successful completion, e.g., MemCo achieves better final outcomes with a step count close to G-Memory's.

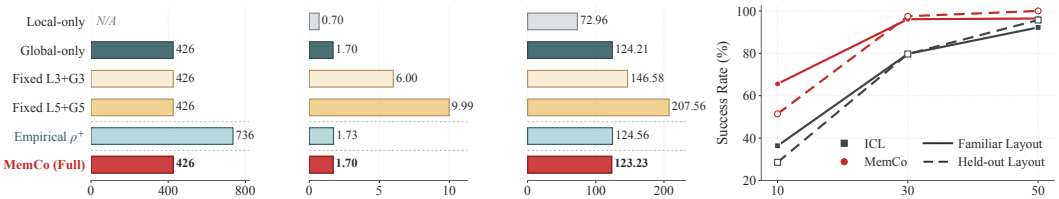
Table 2: Generalization across LLM backbones and to more complex tasks.

(a) FEVER across different LLM backbones.

Method	Claude-Haiku-4.5		Gemini-2.5-flash-lite		DeepSeek-v4-flash	
	Acc. ↑	Steps ↓	Acc. ↑	Steps ↓	Acc. ↑	Steps ↓
ICL	54.24	10.44	49.15	6.24	72.88	**3.02**
A-Mem	62.71	9.02	45.76	5.98	63.56	4.35
MemSkill	60.17	7.46	53.39	6.05	68.65	3.54
MemRL	54.24	7.69	54.24	3.47	67.80	3.67
G-Memory	57.63	7.95	44.92	5.15	–	–
MemCo	**69.49**	**6.38**	**72.04**	**2.42**	**77.97**	3.04

(b) ScienceWorld with Qwen3-32B.

Method	Qwen3-32B			
	Success ↑	Progress ↑	Score ↑	Steps ↓
ICL	13.80	79.80	25.75	10.77
A-Mem	17.33	82.17	25.81	10.87
MemSkill	13.67	81.50	28.13	11.79
MemRL	8.33	77.00	22.25	**5.21**
G-Memory	21.67	**91.33**	30.02	9.67
MemCo	**30.47**	82.90	**34.39**	9.57



(a) Global-Memory Size (b) Retrieved Entries (c) Prompt Tokens (d) Budget Sensitivity

Figure 3: **Memory efficiency and interaction-budget sensitivity on ALFWorld with Qwen3-32B.** Panels (a)–(c) report the global-memory size, average retrieved memory entries, and memory-prompt tokens per retrieval call. Panel (d) compares MemCo and ICL under different interaction budgets.

4.2 Mechanism and Sensitivity Analysis

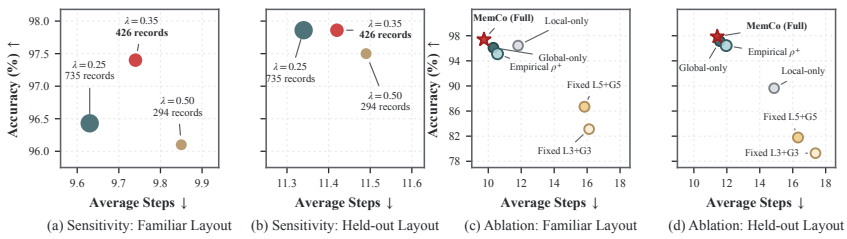
Following the memory lifecycle introduced in Section 3, we conduct controlled studies of MemCo's complementary memory scopes, evidence-aware promotion, and query-guided memory use. Unless otherwise specified, all variants use the same frozen local-memory snapshot and differ only in the component under evaluation. These results represent controlled within-snapshot comparisons rather than independent repetitions of the main experiments.

Complementary Local and Global Memory. Fig. 4(c)-(d) shows that Local-only performs well on familiar layouts but degrades on held-out layouts, whereas Global-only narrows this gap. Held-out accuracy increases from 89.64% with Local-only to 97.14% with Global-only and 97.86% with the full system. Retaining local evidence also improves familiar-layout accuracy over Global-only, from 96.10% to 97.40%. Their adaptive combination achieves the highest overall accuracy with shorter trajectories than either isolated scope, supporting complementary roles for local grounding and transferable global guidance with abstract rules and workflows.

Finite-Evidence-Aware Progressive Collaboration. We replace the Wilson lower confidence bound promotion rule $W(\bar{p})$ with its counterpart $\rho^+(\bar{p})$ while keeping $\lambda = 0.35$ and the frozen local evidence fixed. As shown in Fig. 3(a)-(c) and Fig. 4(c)-(d), Wilson promotion reduces the global bank from 736 to 426 records and improves familiar- and held-out-layout accuracy by 2.27 and 1.43 percentage points, respectively, with fewer interaction steps. Because the two variants inject nearly the same number of entries per decision, the difference is not due to prompt size alone. This result is consistent with Wilson calibration favoring candidates supported by more reliable finite evidence.

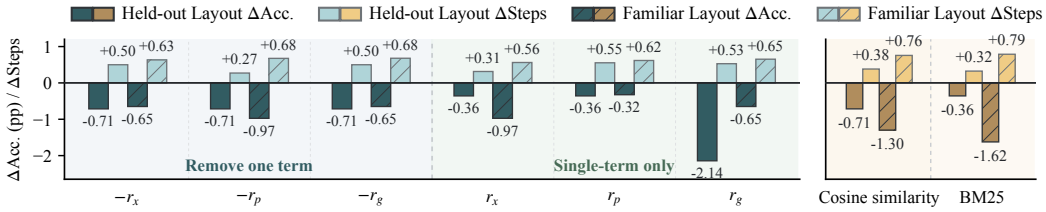
Efficiency under Restricted Interaction Budgets. As shown in Fig. 3(d), MemCo improves when the interaction budget increases from 10 to 30 steps, but gains only marginally from 30 to 50 steps on both splits. This early saturation suggests that a 30-step budget is largely sufficient in this setting for MemCo to exploit its retrieved experience effectively. In comparison, ICL improves more gradually as the budget increases and requires more interaction opportunities to approach MemCo's performance. By varying step limits, MemCo at 30 steps already exceeds ICL at 50 steps on both splits, supporting improved task completion under smaller execution budgets.

Sensitivity to the Promotion Threshold. Fig. 4(a)-(b) evaluates the robustness of the promotion threshold by varying λ while keeping the same frozen local-memory snapshot and reconstructing only the global memory. Increasing λ reduces the global memory from 735 to 294 records, while accuracy and trajectory length remain insensitive. Threshold $\lambda = 0.35$ reduces the number of global memory records by 42.0% compared with $\lambda = 0.25$, while maintaining similar accuracy and interaction steps.



(a) Sensitivity: Familiar Layout (b) Sensitivity: Held-out Layout (c) Ablation: Familiar Layout (d) Ablation: Held-out Layout

Figure 4: **Sensitivity and component ablations on ALFWorld with Qwen3-32B.** Panels (a)–(b) report familiar-layout and held-out-layout performance under different Wilson promotion thresholds λ; bubble area represents the number of promoted global-memory records. Panels (c)–(d) compare memory-scope, fixed-concatenation, empirical-promotion, and full MemCo variants.



(a) Relevance-Scoring Components (b) Retrieval Strategy

Figure 5: **Ablation of relevance scoring and retrieval on ALFWorld with Qwen3-32B.** Bars show changes from full MemCo, with ΔAcc. measured in percentage points and ΔSteps in interactions. In (a), for each relevance facet $d \in \{x, p, g\}$, $-r_d$ removes term r_d, whereas r_d retains only that term. Panel (b) replaces MemCo's retrieval strategy with cosine similarity or BM25.

Query-Guided Memory Support. The fixed-concatenation variants in Fig. 4(c)–(d) test whether MemCo benefits merely from exposing the policy model to more stored experience. As shown in Fig. 4(c)–(d) and Fig. 3(b)–(c), the results show the opposite: L5+G5 trails full MemCo by 10.71 and 16.07 percentage points on the Familiar and Held-out Layout splits, respectively, despite injecting more memory. These results indicate that the way retrieved records are selected and composed matters; simply concatenating more records does not recover the benefit of adaptive composition. The full system uses less context while achieving higher accuracy and shorter trajectories. This is consistent with less targeted memory leading irrelevant guidance into state-specific decision making.

Fig. 5(a) examines the state, phase, and goal relevance facets. Removing any one facet or retaining only a single facet lowers accuracy and increases interaction steps on both splits. The all-zero relevance control in Appendix C also underperforms the full method. These within-snapshot comparisons support the complementary, incremental contributions of the three facets to memory selection. Fig. 5(b) examines an orthogonal design choice: how memory relevance is scored. It replaces MemCo's explicit state, phase, and goal matching with cosine similarity or BM25. Both alternatives underperform the full method in accuracy and efficiency, and neither consistently dominates across splits. These comparisons support explicit state, phase, and goal matching over generic semantic or lexical similarity for selecting memory entries in the evaluated setting.

Qualitative trajectory analysis in Appendix I further illustrates how this mechanism combines transferable global workflows with environment-specific local grounding, while excluding incompatible source-environment guidance in the shown cross-environment trajectories.

5 Conclusion

In this work, we introduced MemCo, a memory-centric collaboration framework for generalizing LLM agents to unseen interactive environments. MemCo constructs local memory from each agent's interaction trajectories, progressively promotes reusable abstract records to shared global memory, and adaptively selects and composes local and global evidence into a decision-relevant memory context during online interaction. Experiments across four heterogeneous benchmarks show that MemCo improves task success and generally reduces redundant exploration compared with memory-free, isolated-memory, and shared-memory baselines.

AI USE STATEMENT

We used generative AI tools for conceptual and methodological feedback, implementation assistance, result interpretation, and manuscript writing improvement. GPT-4o-mini additionally supported FEVER domain labeling (Appendix F.3). The authors take responsibility for the final manuscript, code, and results.

ETHICS STATEMENT

MemCo may reduce redundant exploration by sharing reusable experience across agents. Shared memories may nevertheless propagate erroneous, biased, or malicious content and expose sensitive information. Our benchmark evaluation does not establish privacy guarantees or adversarial robustness. Real-world deployment would require additional access controls, memory auditing, and human oversight.

REPRODUCIBILITY STATEMENT

Section 3 and Appendix B describe local memory construction, role-level abstraction, evidence aggregation and global promotion, and query-guided retrieval and composition. Section 4 reports the evaluation backbones, inference setup, and primary hyperparameters. Appendix F describes benchmark domain construction and task selection, and Appendix G provides the per-step prompt template. The theoretical analysis and statistical procedures are provided in Appendices D and H, respectively. Table 1 reports means and standard deviations over three independent runs, whereas Section 4.2 specifies the fixed-memory protocol used for the controlled component studies.

REFERENCES

Lawrence D Brown, T Tony Cai, and Anirban DasGupta. Interval estimation for a binomial proportion. *Statistical science*, pp. 101–117, 2001.

Zouying Cao, Jiaji Deng, Li Yu, Weikang Zhou, Zhaoyang Liu, Bolin Ding, and Hai Zhao. Remember me, refine me: A dynamic procedural memory framework for experience-driven agent evolution. *arXiv preprint arXiv:2512.10696*, 2025.

Yurui Chang, Yiran Wu, Qingyun Wu, and Lu Lin. Memcollab: Cross-model memory collaboration via contrastive trajectory distillation. *arXiv preprint arXiv:2603.23234*, 2026.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready ai agents with scalable long-term memory. *arXiv preprint arXiv:2504.19413*, 2025.

Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine learning research*, 7(Jan):1–30, 2006.

Ruomeng Ding, Wei Cheng, Minglai Shao, and Chen Zhao. Skillgen: Learning domain skills for in-context sequential decision making. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 30512–30520, 2026.

Runnan Fang, Yuan Liang, Xiaobin Wang, Jialong Wu, Shuofei Qiao, Pengjun Xie, Fei Huang, Huajun Chen, and Ningyu Zhang. Memp: Exploring agent procedural memory. *arXiv preprint arXiv:2508.06433*, 2025.

Sture Holm. A simple sequentially rejective multiple test procedure. *Scandinavian journal of statistics*, pp. 65–70, 1979.

Chengyou Jia, Minnan Luo, Zhuohang Dang, Qiushi Sun, Fangzhi Xu, Junlin Hu, Tianbao Xie, and Zhiyong Wu. Agentstore: Scalable integration of heterogeneous agents as specialized generalist computer assistant. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 8908–8934, 2025.

To Eun Kim, Xuhong He, Dishank Jain, Ambuj Agrawal, Negar Arabzadeh, and Fernando Diaz. Multi-agent transactive memory. *arXiv preprint arXiv:2606.19911*, 2026.

Taeyoon Kwon, Dongwook Choi, Hyojun Kim, Sunghwan Kim, Seungjun Moon, Beong-woo Kwak, Kuan-Hao Huang, and Jinyoung Yeo. Embodied agents meet personalization: Investigating challenges and solutions through the lens of memory utilization. *arXiv preprint arXiv:2505.16348*, 2025.

Weihao Li, Jun Bai, and Ziyang Song. Fedagentke: Federated semantic knowledge evolution for heterogeneous agents. *arXiv preprint arXiv:2607.21361*, 2026.

Sirui Liang, Pengfei Cao, Jian Zhao, Wenhao Teng, Xiangwen Liao, Jun Zhao, and Kang Liu. Learning how to remember: A meta-cognitive management method for structured and transferable agent memory. *arXiv preprint arXiv:2601.07470*, 2026.

Jun Liu, Zhenglun Kong, Changdi Yang, Fan Yang, Tianqi Li, Peiyan Dong, Joannah Nanjekye, Hao Tang, Geng Yuan, Wei Niu, et al. Rcr-router: Efficient role-aware context routing for multi-agent llm systems with structured memory. *arXiv preprint arXiv:2508.04903*, 2025.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the association for computational linguistics*, 12:157–173, 2024.

Chang Ma, Junlei Zhang, Zhihao Zhu, Cheng Yang, Yujiu Yang, Yaohui Jin, Zhenzhong Lan, Lingpeng Kong, and Junxian He. Agentboard: An analytical evaluation board of multi-turn llm agents. *Advances in neural information processing systems*, 37:74325–74362, 2024.

Jiayan Nan, Wenquan Ma, Wenlong Wu, and Yize Chen. Nemori: Self-organizing agent memory inspired by cognitive science. *arXiv preprint arXiv:2508.03341*, 2025.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G Patil, Ion Stoica, and Joseph E Gonzalez. Memgpt: Towards llms as operating systems. *arXiv preprint arXiv:2310.08560*, 2023.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pp. 1–22, 2023.

Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed H Chi, Nathanael Schärli, and Denny Zhou. Large language models can be easily distracted by irrelevant context. In *International Conference on Machine Learning*, pp. 31210–31227. PMLR, 2023.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in neural information processing systems*, 36:8634–8652, 2023.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning. *arXiv preprint arXiv:2010.03768*, 2020.

Xiangru Tang, Tianrui Qin, Tianhao Peng, Ziyang Zhou, Daniel Shao, Tingting Du, Xinming Wei, Peng Xia, Fang Wu, He Zhu, Ge Zhang, Jiaheng Liu, Xingyao Wang, Sirui Hong, Chenglin Wu, Hao Cheng, Chi Wang, and Wangchunshu Zhou. Agent kb: Leveraging cross-domain experience for agentic problem solving. *arXiv preprint arXiv:2507.06229*, 2025.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. Fever: a large-scale dataset for fact extraction and verification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 809–819, 2018.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023.

Ruoyao Wang, Peter Jansen, Marc-Alexandre Côté, and Prithviraj Ammanabrolu. Scienceworld: Is your agent smarter than a 5th grader? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 11279–11298, 2022.

Yu Wang and Xi Chen. Mirix: Multi-agent memory system for llm-based agents. *arXiv preprint arXiv:2507.07957*, 2025.

Edwin B Wilson. Probable inference, the law of succession, and statistical inference. *Journal of the American Statistical Association*, 22(158):209–212, 1927.

Chunlong Wu, Ye Luo, Zhibo Qu, and Min Wang. Meta-policy reflexion: Reusable reflective memory and rule admissibility for resource-efficient llm agent. *arXiv preprint arXiv:2509.03990*, 2025.

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.

Menglin Xia, Xuchao Zhang, Shantanu Dixit, Paramaguru Harimurugan, Rujia Wang, Victor Ruhle, Robert Sim, Chetan Bansal, and Saravan Rajmohan. Memora: A harmonic memory representation balancing abstraction and specificity. *arXiv preprint arXiv:2602.03315*, 2026.

Keyi Xiang, Zeyu Feng, Zhuoyi Lin, Yueming Lyu, Boyuan Shi, Yew-Soon Ong, Ivor Tsang, and Haiyan Yin. Flowsearcher: Synthesizing memory-guided agentic workflows for web information seeking. In *International Conference on Learning Representations*, volume 2026, pp. 149189–149217, 2026.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-mem: Agentic memory for LLM agents. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=FiM0M8gcct.

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. *arXiv preprint arXiv:2603.03296*, 2026.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.

Guibin Zhang, Muxin Fu, Guancheng Wan, Miao Yu, Kun Wang, and Shuicheng Yan. G-memory: Tracing hierarchical memory for multi-agent systems. *arXiv preprint arXiv:2506.07398*, 2025.

Haozhen Zhang, Quanyu Long, Jianzhu Bao, Tao Feng, Weizhi Zhang, Haodong Yue, and Wenya Wang. Memskill: Learning and evolving memory skills for self-evolving agents. *arXiv preprint arXiv:2602.02474*, 2026a.

Shengtao Zhang, Jiaqian Wang, Ruiwen Zhou, Junwei Liao, Yuchen Feng, Zhuo Li, Yujie Zheng, Weinan Zhang, Ying Wen, Zhiyu Li, et al. Memrl: Self-evolving agents via runtime reinforcement learning on episodic memory. *arXiv preprint arXiv:2601.03192*, 2026b.

Yang Zhao, Chengxiao Dai, Yue Xiu, Mengying Kou, Yuliang Zheng, and Dusit Niyato. Shardmemo: Masked moe routing for sharded agentic llm memory. *arXiv preprint arXiv:2601.21545*, 2026.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. Memorybank: Enhancing large language models with long-term memory. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 19724–19731, 2024.

Xizhou Zhu, Yuntao Chen, Hao Tian, Chenxin Tao, Weijie Su, Chenyu Yang, Gao Huang, Bin Li, Lewei Lu, Xiaogang Wang, et al. Ghost in the minecraft: Generally capable agents for open-world environments via large language models with text-based knowledge and memory. *arXiv preprint arXiv:2305.17144*, 2023.

APPENDIX CONTENTS

A Extended Related Work **14**

B Algorithms and Implementation **15**

B.1 Algorithmic Overview . . . 15

B.2 Local Graph Representation and Record Evidence . . . 18

B.3 Progressive Memory Collaboration Details . . . 21

B.4 Query-Guided Memory Support Details . . . 24

B.5 Reported Configuration and Scope of the Details . . . 27

C Extended Experimental Results **28**

C.1 Experimental Reproduction Boundaries . . . 28

C.2 Sensitivity to the Wilson Promotion Threshold . . . 28

C.3 Memory Scope, Promotion, and Query-Guided Composition . . . 28

C.4 Interaction-Budget Sensitivity . . . 29

C.5 Relevance Scoring and Retrieval . . . 29

D Theoretical Analysis **30**

D.1 Wilson Finite-Evidence Correction . . . 30

D.2 Stability of Episode-Level Evidence Aggregation . . . 32

E Memory Types and Local–Global Examples **32**

E.1 Type Definitions and Local–Global Correspondence . . . 33

E.2 From Local Records to Promoted Records . . . 34

E.3 Memory Source, Record Type, and Support Role . . . 34

F Benchmark Domain Construction and Evaluation Scope **34**

F.1 ALFWorld Multi-Domain Environment . . . 34

F.2 PDDL Planning and Domain Partitioning . . . 34

F.3 FEVER Domain Partitioning . . . 34

F.4 ScienceWorld Task-Family Partitioning . . . 35

F.5 Acquisition, Evaluation, and Controlled Comparisons . . . 35

G `MemCo` Per-Step Prompt Template **35**

H Cross-Setting Statistical Significance Analysis **36**

H.1 Matched Evaluation Blocks and Average Ranks . . . 36

H.2 Omnibus Tests Across Methods . . . 37

H.3 Paired Comparisons, Effect Sizes, and Interpretation . . . 38

I Case Study: Adaptive Memory Support and Cross-Environment Generalization **39**

J Visualizations of PDDL Memory Structures **40**

J.1 Global Memory Structure . . . 40

J.2 Local Memory Graphs . . . 42

A Extended Related Work

Individual Experience and Agent Memory. Generative Agents (Park et al., 2023) stores observation streams and derives higher-level reflections, while MemGPT (Packer et al., 2023) manages information beyond the immediate context window. MemoryBank (Zhong et al., 2024), Mem0 (Chhikara et al., 2025), and MIRIX (Wang & Chen, 2025) study long-term memory for conversational, personalized, or multimodal interactions, while MEMENTO (Kwon et al., 2025) evaluates memory utilization for personalized embodied assistance. Long contexts themselves do not guarantee effective reuse, since relevant information may become difficult to access and irrelevant context can distract reasoning (Liu et al., 2024; Shi et al., 2023), motivating explicit storage and retrieval of decision-relevant experience. For interactive agents, a closely related line of work turns environment interaction into reusable decision support. Reflexion (Shinn et al., 2023) stores verbal reflections from previous trials; Voyager (Wang et al., 2023) and GITM (Zhu et al., 2023) accumulate reusable skills or knowledge for open-world interaction; and SkillGen (Ding et al., 2026) extracts domain skills for sequential decision making. Memp (Fang et al., 2025) studies procedural-memory construction, retrieval, and update, while ReMe (Cao et al., 2025) distills and refines procedural experience. Meta-Policy Reflexion (Wu et al., 2025) consolidates reflections into reusable rules, whereas A-Mem (Xu et al., 2025), Nemori (Nan et al., 2025), MemRL (Zhang et al., 2026b), and MemSkill (Zhang et al., 2026a) improve memory organization, episodic reuse, or memory-management policies. These methods primarily improve how experience accumulated within an individual agent or memory system is represented and reused. When multiple agents interact with different environments, a complementary question is how experience acquired by one agent can become useful to others.

Agent Collaboration through Shared Experience. Multi-agent collaboration is commonly realized through communication, agent selection, or execution-time orchestration, e.g., AutoGen (Wu et al., 2024) and AgentStore (Jia et al., 2025). Experience sharing provides a complementary form of collaboration in which experience acquired in prior interactions can support other agents later. Recent work realizes such sharing through different representations. G-Memory (Zhang et al., 2025) organizes prior collaboration within a multi-agent system through hierarchical interaction, query, and insight memory. MATM (Kim et al., 2026) supports population-level storage and retrieval of producer-agent trajectories for later use by consumer agents. Agent KB (Tang et al., 2025) aggregates trajectories from heterogeneous agent frameworks into a structured knowledge base for cross-domain planning and feedback. MemCollab (Chang et al., 2026) distills shared constraints from trajectories generated by different backbone models, while FedAgentKE (Li et al., 2026) evolves transferable semantic knowledge across heterogeneous agents through federated distillation and aggregation. Collectively, these works demonstrate that experience can be reused beyond its original interaction and, in different settings, across agents, models, or agent frameworks. `MemCo` builds on this direction for independently acting agents that accumulate experience in different environments. It keeps environment-specific experience local while sharing reusable records across agents, so that cross-agent experience can be used without collapsing the context in which it was acquired.

Constructing and Using Shared Memory. A first question is how reusable experience should be represented and retained. Procedural-memory methods show that trajectories can be distilled into more compact skills, scripts, rules, or workflows (Ding et al., 2026; Fang et al., 2025; Cao et al., 2025; Wu et al., 2025). FlowSearcher (Xiang et al., 2026) synthesizes memory-guided workflows from historical experience, while PlugMem (Yang et al., 2026) organizes episodic experience into a knowledge-centric memory graph. MCMA (Liang et al., 2026) learns memory structures at different abstraction levels for selective transfer under distribution shift, and Memora (Xia et al., 2026) explicitly balances abstraction against specificity. Together, these studies highlight that transferable memory benefits from capturing reusable structure while retaining information needed for later application. A second question is how stored memory should be used for the current decision. RCR-Router (Liu et al., 2025) dynamically selects structured memory subsets according to agent role and task stage, while ShardMemo (Zhao et al., 2026) uses masked mixture-of-experts routing over sharded memory. These works highlight that the utility of stored memory also depends on context-aware selection at inference time. `MemCo` connects these two stages in a single memory-centric collaboration process: environment-specific evidence is retained locally, reusable records are promoted using evidence accumulated across agents, and local and global memory are selected and grounded according to the receiving agent's current state, progress, goal, and available actions. These mechanisms jointly define the proposed collaboration framework rather than treating shared-memory construction and decision-time use as separate problems.

B Algorithms and Implementation

B.1 Algorithmic Overview

To improve reproducibility, this section summarizes the concrete procedures used by MemCo. Algorithm 1 constructs local graph memory and extracts pattern records from interaction trajectories. Algorithm 4 specifies the structural merge used to consolidate one agent's episodic graphs. Algorithm 2 abstracts the shareable pattern and actionable records in $\mathcal{P}_i^l \cup \mathcal{E}_i^l$, aggregates their evidence across agents, and rebuilds the Wilson-promoted global snapshot. Algorithm 6 separately specifies the exact-key update of promoted records in global memory. Algorithm 3 retrieves and composes local and global memory according to the current decision context.

Following the main text, local memory is $\mathcal{M}_i^l = (G_i^l, \mathcal{P}_i^l, \mathcal{E}_i^l, \mathcal{R}_i^l)$. Here $p \in \mathcal{P}_i^l \cup \mathcal{E}_i^l$ denotes a shareable local pattern or actionable record and $\bar{p} = \mathrm{Abstract}(p)$ its abstract form. The local graph G_i^l and concrete cases and reflections $\mathcal{R}_i^l$ remain local. At decision time, m denotes a memory entry, including a record or a bounded transition entry derived from G_i^l. We use N for the number of agents and N_i for agent i's episodes. The counters $c_i^\omega(p)$ and $n_i^\omega(p)$ denote occurrence-level and episode-reduced evidence, respectively; c_t^h denotes rendered text for support role h.

Table 13 provides a unified reference for the memory components, their record types, local examples, and possible global counterparts. Appendix E distinguishes these stored types from the support roles used during retrieval and composition.

Algorithm 1 Local Memory Construction

Input: Agent i's interaction trajectories $\{\tau_n\}_{n=1}^{N_i}$, where $\tau_n = \{(\boldsymbol{x}_t, \boldsymbol{a}_t, \boldsymbol{f}_t, \boldsymbol{x}_{t+1})\}_{t=1}^{T_n}$
Return: Local memory $\mathcal{M}_i^l = (G_i^l, \mathcal{P}_i^l, \mathcal{E}_i^l, \mathcal{R}_i^l)$
for each trajectory τ_n **do**
  Initialize an episodic graph $G_{\tau_n} = (V_{\tau_n}, E_{\tau_n})$.
  for each transition $(\boldsymbol{x}_t, \boldsymbol{a}_t, \boldsymbol{f}_t, \boldsymbol{x}_{t+1}) \in \tau_n$ **do**
    At the method level, associate state nodes v_t^x, v_{t+1}^x, action node v_t^a, and feedback node v_t^f with the normalized observation, parsed operation, and observed outcome.
    Represent the interaction as $v_t^x \rightarrow v_t^a \rightarrow v_t^f \rightarrow v_{t+1}^x$. In the serialized graph, materialize the $v_t^x \rightarrow v_t^a \rightarrow v_{t+1}^x$ backbone and attach the information represented by v_t^f to the corresponding transition.
    Store the attached feedback through transition statistics and, when present, typed state-delta, subgoal, and failure relations together with the record evidence ledger.
  end for
end for
Preserve the observed transition order within each episodic graph so that record induction follows the executed trajectory, while the merged graph accumulates recurring structure across episodes.
Construct the local graph as $G_i^l \leftarrow \mathrm{LocalMerge}(\{G_{\tau_n}\}_{n=1}^{N_i})$ using Algorithm 4.
Initialize $\mathcal{P}_i^l \leftarrow \emptyset$, $\mathcal{E}_i^l \leftarrow \emptyset$, and $\mathcal{R}_i^l \leftarrow \emptyset$.
for each trajectory τ_n and its episodic graph G_{τ_n} **do**
  Induce workflows, preconditions, repairs, and failure-pattern records from the observed episode structure and add them to $\mathcal{P}_i^l$.
  Extract state–action–outcome relations into $\mathcal{E}_i^l$, and retain concrete examples, failure diagnoses, and recoveries in $\mathcal{R}_i^l$.
end for
Within $\mathcal{P}_i^l$ and $\mathcal{E}_i^l$, consolidate records with the same local record key and union their occurrence references.
for each shareable local record $p \in \mathcal{P}_i^l \cup \mathcal{E}_i^l$ **do**
  Initialize occurrence counters $c_i^+(p), c_i^-(p), c_i^0(p)$ to zero and an episode ledger $\mathcal{J}_i(p) \leftarrow \emptyset$.
  for each observed occurrence j of p **do**
    Assign signed evidence $e_j(p) \in \{+, -\}$ according to whether the observed feedback and transition support or contradict the relation encoded by p.
    if $e_j(p) = +$ **then**
      $c_i^+(p) \leftarrow c_i^+(p) + 1$.
    else
      $c_i^-(p) \leftarrow c_i^-(p) + 1$.
    end if
    if occurrence j has no observable state or task progress **then**
      $c_i^0(p) \leftarrow c_i^0(p) + 1$.
    end if
    Add the signed annotation and optional stall marker to $\mathcal{J}_i(p)[\mathrm{scope}(j), \mathrm{episode}(j)]$.
  end for
  Reduce each annotated episode once: contradiction takes precedence for ordinary relations, while a stall marker supports an anti-pattern. Store the resulting promotion counts as $n_i^+(p), n_i^-(p), n_i^0(p)$.
end for
Return $\mathcal{M}_i^l = (G_i^l, \mathcal{P}_i^l, \mathcal{E}_i^l, \mathcal{R}_i^l)$ according to Eq. 2.

Algorithm 2 Progressive Memory Collaboration and Global Promotion

Input: Shareable local records $\{\mathcal{P}_i^l \cup \mathcal{E}_i^l\}_{i=1}^N$, promotion threshold λ, and one-sided tail probability α
Return: Rebuilt global memory $\mathcal{M}^g$
Initialize an empty global memory $\mathcal{M}^g \leftarrow \emptyset$.
for each $p \in \mathcal{P}_i^l \cup \mathcal{E}_i^l$ from each agent i **do**
    Construct its abstract form $\bar{p} \leftarrow \mathrm{Abstract}(p)$ using Algorithm 5; keep p local if no transferable abstract form is defined.
    Compute its deterministic global record key $\kappa_g(\bar{p})$.
    Merge p into the aggregate for $\kappa_g(\bar{p})$: sum its occurrence statistics, union its source provenance, and register its episode-reduced counts $n_i^+(p)$, $n_i^-(p)$, and $n_i^0(p)$.
end for
for each abstract-record aggregate $\bar{p}_\kappa$ **do**
    Aggregate the episode-reduced counts according to Eq. 3 to obtain $n^+(\bar{p}_\kappa)$, $n^-(\bar{p}_\kappa)$, and diagnostic $n^0(\bar{p}_\kappa)$; occurrence-level counters are not used in the Wilson sample.
    Compute the signed evidence size $s \leftarrow n^+(\bar{p}_\kappa) + n^-(\bar{p}_\kappa)$ and $\rho^+ \leftarrow n^+(\bar{p}_\kappa)/\max(s, 1)$.
    Set $z \leftarrow \Phi^{-1}(1-\alpha)$ and set $W \leftarrow 0$ if $s = 0$; otherwise compute the one-sided Wilson lower bound using Eq. 4.
    if $\bar{p}_\kappa$ passes its record-type structural check, passes the applicable low-information check, and $W \geq \lambda$ **then**
        Update the rebuilt global memory as $\mathcal{M}^g \leftarrow \mathrm{GlobalMerge}(\mathcal{M}^g, \{\bar{p}_\kappa\})$ using Algorithm 6.
    end if
end for
Atomically publish the complete rebuilt snapshot for subsequent queries; do not change the memory context of a decision already in progress.
Return $\mathcal{M}^g$.

Algorithm 3 Query-guided Local–Global Memory Support

Input: Task goal $\boldsymbol{g}$, observation $\boldsymbol{x}_t$, available actions $\mathcal{A}_t$, interaction history $\tau_{<t}$, local memory $\mathcal{M}_i^l$, global memory $\mathcal{M}^g$, and retrieval caps k^l, k^g
Return: Memory context $\mathcal{C}_t = [c_t^{\text{ground}}, c_t^{\text{workflow}}, c_t^{\text{reflection}}]$
Construct the current decision query $\boldsymbol{q}_t \leftarrow \mathcal{Q}(\boldsymbol{g}, \boldsymbol{x}_t, \mathcal{A}_t, \tau_{<t})$ following the query definition in Section 3.5.
Retrieve local memory using $\mathcal{S}_t^l \leftarrow \mathbb{R}(\boldsymbol{q}_t, \mathcal{M}_i^l, k^l)$.
Retrieve global memory using the same function: $\mathcal{S}_t^g \leftarrow \mathbb{R}(\boldsymbol{q}_t, \mathcal{M}^g, k^g)$.
Use Algorithm 7 for both retrieval calls, consistent with Eq. 7.
Define $\mathcal{H} = \{\text{ground}, \text{workflow}, \text{reflection}\}$.
Compose role-specific support as $\{\mathcal{U}_t^h\}_{h\in\mathcal{H}} \leftarrow \mathrm{Compose}\big(\mathcal{S}_t^l, \mathcal{S}_t^g \mid \boldsymbol{q}_t, \mathcal{A}_t\big)$ using Algorithm 8.
Render the selected support as $\mathcal{C}_t \leftarrow \mathrm{Render}\big(\boldsymbol{q}_t, \{\mathcal{U}_t^h\}_{h\in\mathcal{H}}\big)$ using Algorithm 9.
Return $\mathcal{C}_t$.

B.2 Local Graph Representation and Record Evidence

B.2.1 Trajectory and Episodic Graph Representation

Given an interaction episode

$$\tau = \{(\boldsymbol{x}_t, \boldsymbol{a}_t, \boldsymbol{f}_t, \boldsymbol{x}_{t+1})\}_{t=1}^{T}, \tag{11}$$

MemCo normalizes the interaction records and constructs an episodic graph

$$G_\tau = (V_\tau, E_\tau). \tag{12}$$

At the method level, state nodes v_t^x, action nodes v_t^a, and feedback nodes v_t^f represent the normalized observation, parsed operation, and environment feedback, respectively. Directed edges connect them according to the temporal interaction: $v_t^x \to v_t^a \to v_t^f \to v_{t+1}^x$. The feedback node records the observed outcome associated with the executed action, including state progress, failure, or task completion information.

In the serialized implementation, v_t^f is not materialized as a separate stored node for every transition. Instead, its information is attached to the corresponding transition through edge statistics and, when present, typed state-delta, subgoal, and failure relations together with the record evidence ledger. The stored graph backbone is therefore $v_t^x \to v_t^a \to v_{t+1}^x$, while preserving the feedback information represented by v_t^f in the paper-level episodic graph. The structural merge below operates on this stored realization of the same episodic graph; we retain the notation G_τ because the feedback semantics above are preserved in its attached transition fields.

Record induction remains anchored to the observed order of each completed episode and its temporary episodic graph. The persistent merged graph described next is a structural index: it does not retain per-edge episode membership and is not used to reconstruct an episode-consistent trajectory after merging.

B.2.2 Structural Matching and Local Graph Merge

To connect related experience across episodes, MemCo compares graph elements using structural equality after their contents have been normalized. Two nodes are considered equivalent only when both their node types and normalized contents match, while two edges are equivalent only when they share the same relation and equivalent endpoints. Matched elements accumulate their statistics. In the serialized representation, normalized feedback associated with a transition is compared through its attached transition payload and typed relations rather than through a separately stored feedback node. The local memory records the completed episode identifiers at store level, but not a membership list on every merged node or edge. For each agent i, equivalent graph elements across its episodic trajectories are aligned to construct

$$G_i^l = \mathrm{LocalMerge}\left(\{G_{\tau_n}\}_{n=1}^{N_i}\right). \tag{13}$$

The resulting graph therefore provides a connected aggregate structural index across trajectories rather than a collection of isolated action records. An adjacent path in this merged index is not asserted to be a newly observed single-episode trajectory. This operation acts on graph nodes and edges before abstraction; it is not the global exact-key record update in Algorithm 6.

B.2.3 Type-Specific Record Extraction

The local graph exposes recurring structures across episodes, from which MemCo induces reusable records. Record induction remains anchored to the observed order of the contributing episodes, so it does not mine arbitrary synthetic paths from the merged graph. Recurring records with the same normalized content are consolidated so that repeated observations contribute evidence to the same local entry.

Specifically, a *workflow* pattern captures an ordered sequence of progress phases and associated operations; a *precondition* pattern records a temporal transition associated with successful progress; a *repair* pattern records a successful action that resolves an observed failure; and a *failure* pattern records an action associated with a failure or no-progress outcome. Here, a precondition is an

Algorithm 4 LocalMerge: Structural Merge of Episodic Graphs

Input: Episodic graphs $\{G_{\tau_n} = (V_{\tau_n}, E_{\tau_n})\}_{n=1}^{N_i}$ for agent i
Return: Merged local graph $G_i^l = (V(G_i^l), E(G_i^l))$
Initialize $V(G_i^l) \leftarrow \emptyset$, $E(G_i^l) \leftarrow \emptyset$, and an empty map from episodic node identifiers to matching local nodes.
for each episodic graph G_{τ_n} **do**
  for each node $v \in V_{\tau_n}$ **do**
    Form the local node key $\kappa_v(v) = (\text{type}(v), \text{Normalize}(\text{content}(v)))$.
    if no node in $V(G_i^l)$ has key $\kappa_v(v)$ **then**
      Insert a local copy of v into $V(G_i^l)$.
    else
      Reuse the matching local node and accumulate node statistics.
    end if
    Map the episodic identifier of v to that local node.
  end for
  for each edge $e = (u, \eta, v) \in E_{\tau_n}$ **do**
    Resolve u and v to their matching local nodes and form $\kappa_e(e) = (\kappa_v(u), \eta, \kappa_v(v))$.
    if no edge in $E(G_i^l)$ has key $\kappa_e(e)$ **then**
      Insert a local edge with relation η into $E(G_i^l)$.
    else
      Reuse the matching edge and accumulate edge statistics; retain the payload of the stored signature entry.
    end if
  end for
end for
Return $G_i^l = (V(G_i^l), E(G_i^l))$.

experience-derived condition for progress, not a claim that a logically necessary or sufficient operator precondition has been proved.

The other local components in Eq. 2 retain complementary information. The graph G_i^l preserves connected state–action–feedback experience across episodes. Actionable knowledge $\mathcal{E}_i^l$ stores state–action–outcome relations that can ground an executable action, while concrete cases and reflections $\mathcal{R}_i^l$ retain episode-specific examples, failure diagnoses, and recoveries. These components may describe related transitions but serve different purposes: patterns summarize recurring structure, actionable knowledge supports a concrete decision, and $\mathcal{R}_i^l$ keeps instance-level context.

Extracted records with the same record type and normalized content are consolidated into a single entry in $\mathcal{P}_i^l$; actionable records in $\mathcal{E}_i^l$ are consolidated analogously by their local normalized keys. This record consolidation is distinct from graph-element merging and from global promotion. Only selected records in $\mathcal{P}_i^l \cup \mathcal{E}_i^l$ are considered for sharing; G_i^l and $\mathcal{R}_i^l$ remain local. Table 13 summarizes the corresponding memory types and sharing scope.

B.2.4 Record Evidence and Local Counts

Evidence is defined relative to the relation encoded by each shareable record $p \in \mathcal{P}_i^l \cup \mathcal{E}_i^l$. A record may be observed multiple times within one episode or across different episodes. For its j-th occurrence, we assign a signed evidence label

$$e_j(p) \in \{+, -\}, \tag{14}$$

where $e_j(p) = +$ means that the observed transition supports the relation encoded by p, and $e_j(p) = -$ means that it contradicts that relation.

Occurrence-level counters. The occurrence counters summarize all matched observations of the local record:

$$c_i^{\omega}(p) = \sum_j \mathbb{I}[e_j(p) = \omega], \qquad \omega \in \{+, -\}. \tag{15}$$

No-progress behavior is tracked as an auxiliary, potentially overlapping statistic,

$$c_i^{0}(p) = \sum_j \mathbb{I}[\text{NoProgress}(j, p)]. \tag{16}$$

A no-progress occurrence does not by itself determine the sign $e_j(p)$: the same observed outcome can contradict a workflow that claims progress while supporting a failure or anti-pattern whose relation predicts no progress. The counters $c_i^+(p)$, $c_i^-(p)$, and $c_i^0(p)$ are retained as occurrence-level evidence for local record statistics and retrieval-side ranking. They are *not* the sample counts used by the Wilson promotion rule.

Episode-reduced promotion counters. For global promotion, repeated occurrences of the same local record within one episode are first reduced to at most one signed vote. Let $\mathcal{J}_i(p)$ be the evidence ledger keyed by the local record's source scope and episode identifier. A deterministic, relation-aware reducer maps the annotations under each episode key ξ to $\widetilde{e}_\xi(p) \in \{+, -\}$ or to no vote. For ordinary progress relations, a contradiction takes precedence over support within the same episode. For a failure/no-progress relation, an observed stall supports the asserted anti-pattern rather than being treated as successful execution. An unannotated episode contributes no signed vote. The promotion counters are

$$n_i^\omega(p) = \sum_{\xi \in \mathcal{J}_i(p)} \mathbb{I}[\widetilde{e}_\xi(p) = \omega], \qquad \omega \in \{+, -\}. \tag{17}$$

The diagnostic stall count is

$$n_i^0(p) = \sum_{\xi \in \mathcal{J}_i(p)} \mathbb{I}[z_\xi(p) = 1], \tag{18}$$

where $z_\xi(p)$ indicates that episode key ξ contains a no-progress marker. Like $c_i^0(p)$, $n_i^0(p)$ may overlap the signed counts. It is never added to the Wilson sample size.

The two levels therefore have distinct roles:

$$\{e_j(p), \mathrm{NoProgress}(j,p)\}_j \;\to\; (c_i^+, c_i^-, c_i^0, \mathcal{J}_i) \;\to\; (n_i^+, n_i^-, n_i^0) \;\to\; (n^+(\bar{p}), n^-(\bar{p})) \;\to\; \rho^+(\bar{p}),\, W(\bar{p}).$$

The first counters summarize occurrences and may affect retrieval ranking. For promotion, annotations associated with the same local record, source scope, and episode are first reduced to at most one signed vote, producing $n_i^+(p)$ and $n_i^-(p)$. Equation 3 then aggregates these episode-reduced counts over local records that map to the same abstract record. Wilson uses only $s(\bar{p}) = n^+(\bar{p}) + n^-(\bar{p})$ and $\rho^+(\bar{p}) = n^+(\bar{p}) / \max\{s(\bar{p}), 1\}$. Neither the occurrence counters c_i^ω nor the diagnostic count n^0 enters $s(\bar{p})$ or the Wilson formula. Consequently, repeated occurrences of the same local record within one episode do not directly multiply the Wilson evidence.

Record type	Supporting evidence	Contradicting evidence
Workflow	The observed sequence makes the claimed progress and reaches the recorded outcome.	The sequence is applicable but fails to make the claimed progress or reaches an incompatible outcome.
Precondition / actionable relation	The condition–action relation is observed with compatible progress or outcome.	The same relation is applicable but the observed outcome is incompatible with the stored relation.
Repair	The repair follows the relevant failure context and restores progress.	The proposed repair is applicable but does not resolve the failure context.
Failure / anti-pattern	The observed action reproduces the stored failure or no-progress relation.	The action instead produces compatible progress.

Table 3: Record-level evidence used before episode reduction. Evidence is assigned relative to the relation encoded by the record; the same environment outcome can therefore support one relation while contradicting another.

B.2.5 Deterministic Evidence Assignment

Evidence signs are assigned from normalized interaction signals rather than by an additional LLM call. Each environment interface maps its native observations and outcomes to the common information required by the method, including action admissibility, state or task progress, failure, and terminal success. These normalized signals are interpreted relative to the relation encoded by the record to produce $e_j(p)$ and the optional no-progress marker. The evidence semantics, episode reduction, aggregation, and Wilson promotion rule are shared across evaluation environments; only the interface that exposes the normalized state, action, and feedback fields depends on the environment.

B.3 Progressive Memory Collaboration Details

B.3.1 Role-Level Abstraction

Before global aggregation, MemCo applies Algorithm 5 only to selected records $p \in \mathcal{P}_i^l \cup \mathcal{E}_i^l$. The abstraction preserves the relation that makes a record useful—its type, applicability conditions, action order or state–action relation, and outcome—while removing details that identify only the source episode or scene. When a concrete entity has a supported semantic role in the task representation, the stored entity can be replaced by that role; otherwise it remains an applicability constraint that must be matched before use. Records for which no transferable abstract form can be constructed remain local. Neither G_i^l nor $\mathcal{R}_i^l$ is promoted as a global copy.

Type preservation and shared-record fields. Global memory is therefore a collection of promoted abstractions of $\mathcal{P}_i^l$ and $\mathcal{E}_i^l$, not a duplicate of the complete local memory. For a selected record p, $\bar{p} = \mathrm{Abstract}(p)$ retains the fields that define the transferable relation, while source-episode and scene-specific anchors are removed. A promoted entry additionally carries its aggregated evidence, promotion score, and source provenance as specified in Algorithm 2. Abstraction preserves operational conditions rather than turning a conditional relation into an unconditional rule. Consequently, a shared record is still checked and grounded against the receiving agent's current decision context before it is used.

Each abstracted pattern or actionable record is assigned the deterministic global key

$$\kappa_g(\bar{p}) = \mathrm{StoredRecordID}\big(\mathrm{type}(\bar{p}), \mathrm{sig}(\mathrm{relation}(\bar{p}))\big), \tag{19}$$

where $\mathrm{relation}(\bar{p})$ denotes the retained applicability condition, ordered actions or state–action relation, and outcome fields for the record. Evidence counts, display-only summaries, source sets, and removed local anchors do not participate in the identifier. Thus records are merged only when their transferable relation matches exactly after abstraction.

Consider a schematic household episode in which an agent takes a kettle from a counter, fills it at a sink, and heats it on a stove burner. The local trajectory is first converted into a workflow record and is then abstracted into a role-level representation before promotion. Table 4 shows which information is retained and which local anchors are removed.

Field	Value
Local trajectory	go to counter 2; take kettle 1; go to sink 1; fill kettle 1; go to stoveburner 1; heat kettle 1.
Record type	workflow.
Condition	target_object is fillable and heatable.
Action sequence	find(source = water); fill(target_object, water); move(target_object, heating_device); heat(target_object).
Outcome	target_object becomes heated.
Removed local details	kettle 1; counter 2; sink 1; stoveburner 1; local graph-node ids; raw observation strings.
Global workflow fields: metadata	type = workflow; condition = fillable-heatable target_object; outcome = heated target_object.
Global workflow fields: actions	action 1 = find water source; action 2 = fill target; action 3 = move to heating device; action 4 = heat target.

Table 4: Schematic household abstraction example. The action strings illustrate the existing abstraction fields rather than a verbatim benchmark rollout. Concrete entities are removed, while the workflow type, abstract condition, action order, and outcome relation are retained.

Algorithm 5 Abstract: Role-Level Abstraction

Input: Shareable local record $p \in \mathcal{P}_i^l \cup \mathcal{E}_i^l$ and its record-type abstraction schema
Return: Abstract form $\bar{p}$ with attached occurrence and episode evidence, or $\varnothing$ when p remains local
Initialize $\bar{p}$ from the transferable pattern or actionable relation fields of p.
if no transferable abstract form is defined for p **then**
    Return $\varnothing$.
end if
Preserve the record type, applicability condition, action order, and outcome relation defined by p.
Replace a stored entity by a task role when the task representation supports that mapping; otherwise retain it as an applicability constraint.
Remove the designated episode, scene, layout, raw-observation, and local graph-reference fields for this record type.
Preserve the retained operational relation, action order, occurrence statistics, source sets, and source-scoped episode ledger.
Compute $\kappa_g(\bar{p})$ using Eq. 19.
Return $\bar{p}$.

B.3.2 Pre-Promotion Record Checks

The Wilson test is applied only to records that first pass two deterministic checks. Let $A_{\mathrm{str}}(\bar{p})$ denote structural completeness and $A_{\mathrm{info}}(\bar{p})$ denote the low-information check. A record proceeds to Wilson scoring only when $A_{\mathrm{str}} = A_{\mathrm{info}} = 1$; all shareable record types then use the same Wilson formula and threshold λ.

Record relation	Required transferable content
Workflow	A nonempty applicability condition or task role, an ordered action or phase sequence, and an outcome.
Precondition	A condition, an action relation, and evidence of compatible progress.
Repair / failure relation	A triggering failure or no-progress condition and the corresponding repair or avoidance relation.
Actionable knowledge	A usable condition–action–outcome relation that can be bound to the receiving decision context.

Table 5: Structural completeness checks applied before Wilson promotion. The checks retain only records with sufficient transferable content; incomplete, source-bound, or uninformative records remain local.

The low-information check removes records whose abstract form contains no usable condition, relation, or outcome beyond generic interaction advice. These checks operate before evidence calibration and do not change the Wilson score or threshold. Source provenance and retrieval relevance are retained for later aggregation or decision-time use rather than used as alternative promotion thresholds.

B.3.3 Global Exact-Key Merge of Promoted Records

Global construction does not use embedding similarity or an LLM to decide whether two records are equivalent. It first constructs the applicable abstract form for each selected $p \in \mathcal{P}_i^l \cup \mathcal{E}_i^l$ and then groups the resulting pattern and actionable representations by the key in Eq. 19. Evidence counts do not participate in identifier matching. The Wilson implementation rebuilds each snapshot from all current local memories, so occurrence statistics and source sets are recomputed from those local records rather than cumulatively added to an earlier global report. Episode reduction is completed before global aggregation, and Eq. 3 then sums the resulting episode-reduced signed counts for records with the same exact abstract key. Occurrence counts are therefore never substituted for the promotion counts. The accompanying diagnostic is aggregated as

$$n^0(\kappa) = \sum_{i \in [N],\, p \in \mathcal{P}_i^l \cup \mathcal{E}_i^l : \kappa_g(\mathrm{Abstract}(p)) = \kappa} n_i^0(p), \tag{20}$$

where κ is the global record key. Only $n^+(\kappa) + n^-(\kappa)$ enters the Wilson sample size. This is the same episode-reduction-then-aggregation order specified by Eq. 3.

After aggregation, each record is evaluated by the structural, low-information, and Wilson gates. Because every rebuild starts from an empty global memory, a record that no longer passes is absent from the new snapshot; every retained global record therefore satisfies the promotion rule under the current local evidence.

The exact-record-key rule prevents overly broad merging. A filling workflow and a cleaning workflow may both mention a water source, but they are not the same pattern entry. Tables 6 and 7 retain concrete and global examples for readability and illustrate normalized workflow-identifier matching rather than surface semantic overlap.

Field	Kitchen record	Bathroom record
Concrete entities	cloth, kitchen sink, countertop.	soap, bathroom basin, vanity.
Record type	workflow.	workflow.
Condition after abstraction	cleanable target_object.	cleanable target_object.
Action sequence after abstraction	find water source; clean target_object; place in goal_destination.	find water source; clean target_object; place in goal_destination.
Outcome after abstraction	stored clean target_object.	stored clean target_object.

Merge decision. Merged as one promoted workflow record in $\mathcal{M}^g$. The two $\mathcal{P}_i^l$ records have the same κ_g after induction-time role normalization. Snapshot construction recomputes the source set and evidence totals from all current local records.

Table 6: Positive merge example. Different local objects can map to the same global workflow when the record fields match exactly.

Field	Filling workflow	Cleaning workflow
Condition	fillable target_object.	cleanable target_object.
Action sequence	find water source; fill target_object.	find water source; clean target_object.
Outcome	filled target_object.	clean target_object.

Merge decision. Not merged. The entries share the phrase *water source*, but their normalized workflow identifiers differ because their action sequences and outcomes differ.

Table 7: Negative merge example. Surface overlap is insufficient for merging.

Algorithm 6 implements $\mathrm{GlobalMerge}$ in Eq. 6. Evidence is aggregated before promotion; this update stores the current totals under the appropriate global record key. In contrast, $\mathrm{LocalMerge}$ acts on graph elements (Algorithm 4).

Algorithm 6 $\mathrm{GlobalMerge}$: Exact-Key Update of Promoted Global Records

Input: Global memory $\mathcal{M}^g$ and abstract aggregates $\bar{p}$ that pass the pre-promotion checks and satisfy $W(\bar{p}) \geq \lambda$, carrying the aggregated evidence from Eq. 3
Return: Updated global memory $\mathcal{M}^g$
for each promoted aggregate $\bar{p}$ **do**
  Compute its global record key $\kappa_g(\bar{p})$ using Eq. 19.
  if no stored record has key $\kappa_g(\bar{p})$ **then**
    Insert $\bar{p}$, its current source set, promotion score, and aggregated evidence into $\mathcal{M}^g$.
  else
    Keep one record for the matching global key and replace its mutable fields with the current source set, promotion score, occurrence statistics, and episode-reduced counts $n^+(\bar{p})$, $n^-(\bar{p})$, and $n^0(\bar{p})$.
  end if
end for
Return $\mathcal{M}^g$.

Snapshot realization. Algorithm 2 constructs the global snapshot from the current local memories by reducing episode evidence, grouping records by the exact global key κ_g, and applying the pre-promotion and Wilson checks. A complete rebuilt snapshot is then made available to subsequent decisions. The acquisition and evaluation lifecycle is summarized in Appendix F.5.

B.4 Query-Guided Memory Support Details

B.4.1 Local Graph Query Anchoring

The local graph is queried as a bounded structural index. State anchors are matched to the current query using normalized state content together with location, visible or held entities, task progress, and goal-related cues when available. From each retained anchor, retrieval follows only a bounded local neighborhood of the observed state–action–feedback–next-state transition. In the serialized graph, this is the corresponding state–action–next-state backbone together with its attached feedback fields. The resulting transition entries summarize concrete local evidence that can support the current decision; retrieval does not search arbitrary paths or construct a new long workflow from the merged graph.

Graph-derived entries and records from $\mathcal{P}_i^l$, $\mathcal{E}_i^l$, and $\mathcal{R}_i^l$ enter the same bounded local selection stage. Their graph provenance can strengthen a record when the retrieved state, action, or outcome matches the current query, while incompatible or stale context lowers its priority. All such entries remain subject to the common state, progress, goal, and action-compatibility checks below.

B.4.2 Memory Entry Construction, Matching, and Ranking

At each decision step, `MemCo` constructs the query defined in Section 3.5 from the current goal, observation, available actions, and interaction history. The same retrieval function $\mathbb{R}$ is called independently on local and global memory according to Eq. 7. Local retrieval draws from the local records and bounded graph neighborhoods described above, whereas global retrieval draws from promoted abstract records in $\mathcal{M}^g$.

Candidate entries are evaluated through the three complementary relevance facets in Eq. 8:

$$r(m, \boldsymbol{q}_t) = r_x(m, \boldsymbol{q}_t) + r_p(m, \boldsymbol{q}_t) + r_g(m, \boldsymbol{q}_t).$$

Here r_x measures consistency with the current state and available actions, r_p measures compatibility with task progress, and r_g measures alignment with the goal and its functional roles. These facets score the same memory entry from complementary views; they are not independent pass/fail gates. Each facet is computed by deterministic comparisons over the normalized fields available to the entry: compatible matches contribute positively, explicit mismatches reduce the score, and unavailable optional fields contribute no update. The reported outer weights of the three facets are one. These comparisons use the common normalized representation rather than a benchmark-specific retrieval objective.

Facet	Main emphasis	Examples of matching evidence
r_x	Current state and action context	State or location consistency, visible or held entities, applicable conditions, and compatibility with currently available actions.
r_p	Task and execution progress	Task stage, completed or remaining subgoals, failure context, and whether the stored procedure is appropriate at the current decision phase.
r_g	Goal and functional roles	Alignment between stored object, destination, tool, or action roles and the current task goal.

Applicability constraints are handled before ranking: an entry is removed when a required condition contradicts the current state or when a required action cannot be matched to $\mathcal{A}_t$. These hard compatibility checks are distinct from the soft preferences used to rank the remaining entries.

The joint relevance score is then combined with stored evidence, current action binding, source-specific context, and soft conflict penalties to obtain the full rank used for bounded selection:

$$s_{\mathrm{rank}}(m, \boldsymbol{q}_t) = r(m, \boldsymbol{q}_t) + \Delta_{\mathrm{ev}}(m) + \Delta_{\mathrm{bind}}(m, \boldsymbol{q}_t) + \Delta_{\mathrm{src}}(m, \boldsymbol{q}_t) - p_{\mathrm{soft}}(m, \boldsymbol{q}_t).$$

Here Δ_{ev} summarizes the occurrence-level evidence retained with the entry, Δ_{bind} reflects compatibility between the stored operation or semantic roles and the currently available actions, Δ_{src} incorporates source-specific supporting context (such as query-matched local graph context for local entries or retained applicability and role context for global entries), and p_{soft} collects non-disqualifying conflicts such as stale or weakly mismatched guidance. These terms are deterministic adjustments over the same normalized fields used by the common retrieval interface. Occurrence-level statistics can contribute to retrieval ranking, whereas the episode-reduced counts n^+, n^- are reserved for Wilson promotion. The Wilson score therefore controls membership in global memory rather than being added directly to the online relevance score.

B.4.3 Bounded Selection

Algorithm 7 combines source-aware pool construction, compatibility filtering, joint relevance, and bounded selection. The caps k^l and k^g bound the retrieved sets. Before rank-based filling, selection preserves at most one strongest supporting match for each relevance facet, when such a match is available, so that one dominant score does not remove complementary state-, progress-, or goal-related support. A supporting facet match is one with $r_d(m, \boldsymbol{q}_t) > 0$ for $d \in \{x, p, g\}$. Remaining positions are filled by decreasing full rank. Composition may subsequently reject or deduplicate selected entries after current-environment grounding.

Algorithm 7 $\mathbb{R}$: Source-Aware Retrieval, Filtering, and Bounded Selection

Input: Query $\boldsymbol{q}_t$, one memory $\mathcal{M} \in \{\mathcal{M}_i^l, \mathcal{M}^g\}$, and a nonnegative cap k
Return: Selected memory entries $\mathcal{S}$
Initialize the retrieval pool $\mathcal{B} \leftarrow \emptyset$.
if $\mathcal{M} = \mathcal{M}_i^l = (G_i^l, \mathcal{P}_i^l, \mathcal{E}_i^l, \mathcal{R}_i^l)$ **then**
    Add records from $\mathcal{P}_i^l$, $\mathcal{E}_i^l$, and $\mathcal{R}_i^l$ to $\mathcal{B}$.
    Add bounded transition entries from query-matched neighborhoods of G_i^l, preserving the paper-level state–action–feedback–next-state semantics and the attached serialized feedback.
else
    Add the promoted abstract records in $\mathcal{M}^g$ to $\mathcal{B}$.
end if
Initialize the eligible scored pool $\mathcal{V} \leftarrow \emptyset$.
for each memory entry $m \in \mathcal{B}$ **do**
    Compute $r_x(m, \boldsymbol{q}_t)$, $r_p(m, \boldsymbol{q}_t)$, and $r_g(m, \boldsymbol{q}_t)$.
    if m contradicts a required current condition or a required action cannot be matched to $\mathcal{A}_t$ **then**
        Exclude m from $\mathcal{V}$.
    else
        Compute the full rank from joint relevance, stored evidence, current action binding, source context, and soft conflicts.
        Add m and its facet scores, full rank, provenance, and available action/graph evidence to $\mathcal{V}$.
    end if
end for
Deduplicate equivalent entries in $\mathcal{V}$, keeping the higher-ranked applicable entry, and sort by decreasing full rank.
if $|\mathcal{V}| \leq k$ **then**
    Return $\mathcal{V}$.
end if
Initialize $\mathcal{S} \leftarrow [\,]$.
for each relevance facet $d \in \{x, p, g\}$ **do**
    if $|\mathcal{S}| < k$ and an unselected entry satisfies $r_d(m, \boldsymbol{q}_t) > 0$ **then**
        Append the unselected entry with the largest r_d, breaking ties by full rank.
    end if
end for
for each remaining entry in decreasing full-rank order **do**
    if $|\mathcal{S}| < k$ **then**
        Append the entry to $\mathcal{S}$ without duplication.
    end if
end for
Retain provenance, facet scores, full rank, and action/graph evidence for composition.
Return $\mathcal{S}$.

B.4.4 COMPOSITION AND RENDERING

Retrieval and composition serve different roles. Retrieval selects relevant entries within each memory source, whereas composition determines how the selected local and global entries should be used in the current decision. With $\mathcal{H} = \{\text{ground}, \text{workflow}, \text{reflection}\}$, grounding uses current-state facts and graph relations; workflow support uses plans, procedures, and preconditions; and reflection uses failure patterns, repairs, and blocked actions. Entries whose content is incompatible with the current role or decision context are removed.

For a global record, composition first binds its abstract object, destination, tool, and action roles to the current query and available actions. A global record is not emitted when a required binding cannot be established. Local state constraints and concrete local grounding take precedence over an incompatible abstract recommendation. The remaining support is deduplicated, organized by role, and kept within a bounded context budget before rendering. This stage therefore converts shared knowledge into current-environment support rather than directly copying a stored global record into the prompt.

The renderer serializes the admitted support into the three memory-bearing fields in Eq. 10. It keeps only information needed for the current decision and uses the current bindings for grounded global records rather than source-environment anchors. The full prompt template is shown in Appendix G.

Algorithm 8 Compose: Query-Conditioned Local–Global Support Composition

Input: Selected memory entries $\mathcal{S}_t^l, \mathcal{S}_t^g$, query $\boldsymbol{q}_t$, actions $\mathcal{A}_t$, roles $\mathcal{H} = \{\text{ground}, \text{workflow}, \text{reflection}\}$, and the configured context budget
Return: Role-specific supports $\{\mathcal{U}_t^h\}_{h\in\mathcal{H}}$
Initialize $\mathcal{U}_t^h \leftarrow [\,]$ for every $h \in \mathcal{H}$.
for each memory entry $m \in \mathcal{S}_t^l \cup \mathcal{S}_t^g$ in retained full-rank order **do**
    Preserve the source, full rank, facet scores, and available action/graph evidence attached to m.
    if m is an abstract global record **then**
        Bind its object, destination, tool, and action roles against $\boldsymbol{q}_t$ and $\mathcal{A}_t$.
        if a required role cannot be grounded or its required action is unavailable **then**
            Exclude m from the current composition.
        end if
    end if
    if m remains eligible after binding and compatibility checks **then**
        Determine the compatible role set $\mathcal{H}_m \subseteq \mathcal{H}$ from the record type and its current contextual trigger.
        Assign current-state facts and graph relations to ground; plans, procedures, and preconditions to workflow; and failures, repairs, and blocked actions to reflection whenever the corresponding role is in $\mathcal{H}_m$.
    end if
end for
Remove duplicate support items within each role while retaining the higher-ranked applicable item and its provenance.
Resolve action conflicts by discarding unavailable actions and by letting current-state constraints and concrete local grounding override an incompatible abstract recommendation.
Apply the fixed context budget to the remaining support in deterministic full-rank order.
Retain the admitted memory-bearing support in the corresponding $\mathcal{U}_t^h$ in deterministic rank order.
Return $\{\mathcal{U}_t^h\}_{h\in\mathcal{H}}$.

Algorithm 9 Render: Role-Specific Memory Context Rendering

Input: Query $\boldsymbol{q}_t$ and composed supports $\{\mathcal{U}_t^h\}_{h\in\mathcal{H}}$
Return: Memory context $\mathcal{C}_t = [c_t^{\text{ground}}, c_t^{\text{workflow}}, c_t^{\text{reflection}}]$
for each support role $h \in (\text{ground}, \text{workflow}, \text{reflection})$ **do**
    Order $\mathcal{U}_t^h$ by the deterministic composition order.
    Convert each selected memory entry into a concise instruction using only fields relevant to $\boldsymbol{q}_t$ and role h.
    For grounded global records, emit the current concrete bindings rather than removed scene-specific anchors.
    Deduplicate equivalent instructions and retain the highest-priority applicable instruction for each emitted memory line.
    Store the resulting text in c_t^h; set c_t^h to the empty string when $\mathcal{U}_t^h$ is empty after composition.
end for
If all three rendered fields are empty, return an empty memory context.
Set $\mathcal{C}_t \leftarrow [c_t^{\text{ground}}, c_t^{\text{workflow}}, c_t^{\text{reflection}}]$.
Return $\mathcal{C}_t$.

B.4.5 Concrete Retrieval Example

Suppose the current task is to clean a bar of soap and place it in a vanity. The current observation mentions a bathroom basin, and the environment exposes an admissible action for cleaning the soap with the basin. The local record in this illustrative example is actionable knowledge in $\mathcal{E}_i^l$; the shared record is a promoted workflow from $\mathcal{M}^g$. They support the same decision in different ways: the local relation grounds an executable cleaning action, while the global workflow supplies the clean-before-place ordering. The global roles are bound to the current soap, basin, and vanity before use. The rejected kitchen-specific instruction illustrates a grounding conflict, not an additional observed record in the global snapshot.

Field	Value
Current goal	Clean soap and put it in the vanity.
Current observation	Soap and a basin are visible in the bathroom.
Admissible action	`clean soap with basin`.
Local type and record	Actionable knowledge, $\mathcal{E}_i^l$: the basin can clean the soap; the action relation matches the currently admissible cleaning action.
Local use	Current-state and action compatibility support c_t^{ground}.
Global type and record	Promoted workflow, $\mathcal{M}^g$: for a cleanable target, find a water source; clean `target_object`; place it in `goal_destination`. Outcome: a stored clean target.
Current role bindings	`target_object` $\mapsto$ soap; water source $\mapsto$ basin; `goal_destination` $\mapsto$ vanity.
Global use	Goal-role and task-progress compatibility support c_t^{workflow}; the ordering does not replace local grounding.
Rejected memory entry	`go to kitchen sink`.
Rejection reason	The concrete kitchen location cannot be bound to the current bathroom observation and available actions; the instruction is not used.

Table 8: Schematic typed retrieval and composition example. Local actionable knowledge supplies executable grounding, while a global workflow supplies action order. Source, record type, and support role are shown separately.

B.5 Reported Configuration and Scope of the Details

The reported configuration uses $\alpha = 0.05$, $\lambda = 0.35$, retrieval caps $k^l = k^g = 5$, and unit outer weights for the three relevance facets. These are the paper-level parameters governing promotion and bounded retrieval. Across the evaluated benchmarks, the same paper-level pipeline is used for local memory construction, abstraction and promotion, query-guided retrieval, and composition. Environment interfaces provide the normalized state, action, feedback, goal, progress, and supported task-role fields consumed by this pipeline; benchmark-specific serialization details are not additional

components of the method. The paper-level memory context is the three-role $\mathcal{C}_t$ in Eq. 10; surrounding task-wrapper context shown in Appendix G is kept separate from this local/global retrieval interface.

C Extended Experimental Results

This section provides the numerical results underlying the mechanism and sensitivity analyses in Figs. 3, 4, and 5. Table 9 reports the Wilson-threshold sweep, Table 10 combines the memory-scope, promotion, composition, and memory-cost results, Table 11 reports interaction-budget sensitivity, and Table 12 gives the absolute relevance-scoring and retrieval results.

Unless otherwise specified, the component, promotion, and retrieval variants reuse the same frozen local-memory snapshot and differ only in the evaluated component. These are controlled within-snapshot comparisons rather than independent repetitions of the main experiments. The interaction-budget sweep follows the budget-specific evaluation protocol used in Fig. 3(d). Accuracy and success rates are reported in percent, and Steps denotes the average number of environment interactions. Familiar and Held-out refer to ALFWorld's official `valid_seen` and `valid_unseen` layout splits, respectively. Collectively, the controlled studies examine the three mechanisms emphasized by the method: complementary local/global memory scope, evidence-aware global promotion, and query-guided use of memory for the current decision.

C.1 Experimental Reproduction Boundaries

The main results and the controlled component studies serve different purposes. Table 1 reports mean $\pm$ standard deviation over three runs. The component, promotion, and retrieval studies in this appendix instead reuse a fixed local-memory snapshot so that the evaluated mechanism can be changed without recollecting experience. For threshold sensitivity, only the promoted global snapshot is reconstructed from the same local evidence; for retrieval and composition studies, the underlying memories remain fixed.

The main experiments use the shared setting $\lambda = 0.35$ across benchmarks, while Table 9 reports a controlled sensitivity sweep around this operating point. The cross-setting significance analysis in Appendix H uses the displayed main-result means as matched backbone–task blocks; it does not treat the repeated runs as additional independent blocks.

C.2 Sensitivity to the Wilson Promotion Threshold

Table 9 reconstructs global memory at each threshold while reusing the same local evidence. Increasing λ from 0.25 to 0.50 reduces the global bank from 735 to 294 records. Over this range, familiar-layout accuracy remains between 96.10% and 97.40%, and held-out-layout accuracy between 97.50% and 97.86%. The reported setting $\lambda = 0.35$ uses 426 records, 42.0% fewer than $\lambda = 0.25$, while attaining the highest familiar-layout accuracy and tying the highest held-out-layout accuracy. The lower threshold nevertheless has the fewest steps on both splits. The table therefore characterizes the accuracy–compactness–efficiency trade-off rather than establishing that one threshold optimizes every metric.

Table 9: Sensitivity to the Wilson promotion threshold on ALFWorld with Qwen3-32B. All thresholds use the same frozen local-memory snapshot; only global memory is reconstructed.

λ	Global records	Familiar Acc. (%) ↑	Held-out Acc. (%) ↑	Familiar Steps ↓	Held-out Steps ↓
0.25	735	<u>96.43</u>	**97.86**	**9.63**	**11.34**
0.35	426	**97.40**	**97.86**	<u>9.74</u>	<u>11.42</u>
0.50	294	96.10	<u>97.50</u>	9.85	11.49

C.3 Memory Scope, Promotion, and Query-Guided Composition

Table 10 separates the roles of the two memory sources, the promotion criterion, and the way retrieved records are used. Local-only retains strong familiar-layout accuracy but has a larger familiar–held-

out layout gap; Global-only improves held-out-layout accuracy from 89.64% to 97.14% relative to Local-only. The complete system improves on both isolated sources and uses fewer steps on both splits. Simply concatenating local and global records does not recover this benefit: relative to fixed L5+G5, the full method gains 10.71 and 16.07 percentage points on familiar- and held-out-layout accuracy, respectively.

The empirical-promotion row changes the admission score from $W(\bar{p})$ to $\rho^{+}(\bar{p})$ at the same $\lambda = 0.35$, leaving the structural eligibility checks and query-guided retrieval policy unchanged. Wilson promotion improves accuracy by 2.27 and 1.43 percentage points and reduces steps by 0.82 and 0.54 on the familiar-layout and held-out-layout splits, respectively. Together with Figure 3, this comparison separates the quality of the promoted bank from merely retrieving more records. These within-snapshot results support the contribution of each component without attributing every change to a single underlying cause.

Table 10: **Ablation of memory organization, global promotion, and query-guided composition on ALFWorld with Qwen3-32B.** All variants use the same frozen local-memory snapshot. "Global records" denotes the size of promoted global memory. "Avg. #Mem" and "Avg. Mem. Tok." denote the average number of entries and memory-prompt tokens injected per retrieval call, respectively, averaged over the Familiar and Held-out splits. A dash indicates that no global memory is used.

Setting	Global records	Accuracy (%) ↑		Steps ↓		Avg. #Mem ↓	Avg. Mem. Tok. ↓
		Familiar	Held-out	Familiar	Held-out		
Memory scope and composition policy							
Local-only	–	96.43	89.64	11.81	14.87	0.70	72.96
Global-only	426	96.10	97.14	10.31	11.57	1.70	124.21
Local+Global, fixed concat (L3+G3)	426	83.12	79.29	16.12	17.38	6.00	146.58
Local+Global, fixed concat (L5+G5)	426	86.69	81.79	15.84	16.32	9.99	207.56
Global-memory promotion criterion							
Empirical support ratio ρ^{+}	736	95.13	96.43	10.56	11.96	1.73	124.56
MemCo (Wilson + query-guided)	426	97.40	97.86	9.74	11.42	1.70	123.23

C.4 Interaction-Budget Sensitivity

Table 11 reports the exact success rates underlying Fig. 3(d). This experiment varies only the maximum interaction budget under its budget-specific evaluation protocol.

Table 11: **Sensitivity to the maximum interaction budget on ALFWorld with Qwen3-32B.** Success rates are reported for the Familiar and Held-out layout splits.

Method	Budget	Familiar	Held-out
ICL	10	36.36	28.57
	30	79.72	79.69
	50	92.21	95.71
MemCo	10	65.58	51.43
	30	96.10	97.50
	50	96.43	100.00

C.5 Relevance Scoring and Retrieval

Table 12 reports the absolute results underlying Fig. 5. The main-text figure visualizes signed changes from the full MemCo row, whereas the table reports the corresponding absolute accuracy and trajectory length.

Table 12: **Absolute results for relevance-scoring and retrieval ablations on ALFWorld with Qwen3-32B.** In the component rows, $-r_i$ removes term r_i, whereas an isolated r_i retains only that term. The all-zero control is included for completeness but omitted from the main-text visualization.

Setting	Accuracy (%) ↑		Steps ↓	
	Familiar	Held-out	Familiar	Held-out
Reference configuration				
Full MemCo ($r_x + r_p + r_g$)	97.40	97.86	9.74	11.42
Remove one relevance term				
$-r_x$ (retain $r_p + r_g$)	96.75	97.14	10.37	11.92
$-r_p$ (retain $r_x + r_g$)	96.43	97.14	10.42	11.69
$-r_g$ (retain $r_x + r_p$)	96.75	97.14	10.42	11.92
Single-term relevance				
r_x only	96.43	97.50	10.30	11.74
r_p only	97.08	97.50	10.36	11.98
r_g only	96.75	95.71	10.39	11.95
Additional control				
$r = 0$	96.43	97.14	10.37	11.96
Alternative retrieval strategy				
Cosine similarity	96.10	97.14	10.50	11.80
BM25	95.78	97.50	10.53	11.75

D Theoretical Analysis

This section analyzes two properties of MemCo's global promotion mechanism. We first prove the finite-evidence correction stated in Theorem 3.1 and derive sharper asymptotic and threshold implications. We then characterize the stability of the episode-level evidence aggregation used before Wilson promotion. Throughout this section, n^+ and n^- denote the episode-reduced promotion counts defined in Appendix B.2.4, rather than the occurrence-level statistics used for retrieval. Wilson promotion uses only these signed episode-reduced counts through $s = n^+ + n^-$ and $\rho^+ = n^+/s$ when $s > 0$; neither c^ω nor the diagnostic n^0 enters the Wilson score.

D.1 Wilson Finite-Evidence Correction

Proof of Theorem 3.1. For brevity, let $s = s(\bar{p})$, $\rho = \rho^+(\bar{p})$, and $W = W(\bar{p})$. The Wilson lower bound in Eq. 4 is equivalently the smaller root of

$$(s + z^2)W^2 - (2s\rho + z^2)W + s\rho^2 = 0. \tag{21}$$

Indeed, solving Eq. 21 for W recovers exactly the closed form in Eq. 4.

Rearranging Eq. 21 gives

$$s(\rho - W)^2 = z^2 W(1 - W). \tag{22}$$

Since W is the lower endpoint and therefore $W \le \rho$,

$$\epsilon_W(\bar{p}) = \rho^+(\bar{p}) - W(\bar{p}) = z\sqrt{\frac{W(\bar{p})(1 - W(\bar{p}))}{s(\bar{p})}}. \tag{23}$$

For $\rho^+(\bar{p}) = 0$, Eq. 4 gives $W(\bar{p}) = 0$, so the same identity holds trivially.

Finally, since $0 \le W(\bar{p}) \le 1$ and $W(\bar{p})(1 - W(\bar{p})) \le 1/4$,

$$0 \le \epsilon_W(\bar{p}) \le \frac{z}{2\sqrt{s(\bar{p})}}. \tag{24}$$

The bound is independent of $\rho^+(\bar{p})$, establishing the uniform $O(s(\bar{p})^{-1/2})$ correction stated in Theorem 3.1. □

Corollary D.1 (Exact promotion threshold and required evidence)**.** *For $s = s(\bar{p}) > 0$, $z > 0$, and $0 < \lambda < 1$, the Wilson gate satisfies*

$$W(\bar{p}) \geq \lambda \quad \Longleftrightarrow \quad \rho^+(\bar{p}) \geq \lambda + z\sqrt{\frac{\lambda(1-\lambda)}{s(\bar{p})}}. \tag{25}$$

Equivalently, for integer signed evidence counts,

$$n^+(\bar{p}) \geq \left\lceil s(\bar{p})\lambda + z\sqrt{s(\bar{p})\lambda(1-\lambda)} \right\rceil. \tag{26}$$

For a fixed empirical rate $\rho^+(\bar{p}) > \lambda$, this is equivalent to

$$s(\bar{p}) \geq \frac{z^2\lambda(1-\lambda)}{(\rho^+(\bar{p}) - \lambda)^2}. \tag{27}$$

No finite positive evidence size passes when $\rho^+(\bar{p}) \leq \lambda$. At the all-supporting boundary $n^-(\bar{p}) = 0$, the minimum integer evidence size is

$$s_{\min} = \left\lceil \frac{z^2\lambda}{1-\lambda} \right\rceil. \tag{28}$$

Below this size, no supporting rate can pass the Wilson gate. These conditions concern the Wilson gate only; structural and information eligibility are still required for global promotion.

Proof. Write $\rho = \rho^+(\bar{p})$ and define $Q_\rho(w) = s(\rho - w)^2 - z^2 w(1-w)$. By Eq. 21, its smaller root is W. For $0 < \rho < 1$, $Q_\rho(\rho) < 0$, so ρ lies strictly between the two roots. Consequently, when $\lambda \leq \rho$, the condition $W \geq \lambda$ is equivalent to $Q_\rho(\lambda) \geq 0$. Taking the nonnegative square root gives $\rho - \lambda \geq z\sqrt{\lambda(1-\lambda)/s}$. When $\lambda > \rho$, both sides of the claimed equivalence are false because $W \leq \rho$. The boundary $\rho = 0$ gives $W = 0$; at $\rho = 1$, $W = s/(s + z^2)$ gives the same equivalence directly. Multiplying by s and using integer n^+ proves Eq. 26; rearranging for $\rho > \lambda$ proves Eq. 27. Finally, the right-hand side of Eq. 25 must not exceed one. This is equivalent to $s \geq z^2\lambda/(1-\lambda)$, which is attained when all votes are supporting. □

Pointwise refinement. The uniform bound in Theorem 3.1 controls the largest possible finite-evidence correction over supporting rates. The dependence on a fixed supporting rate can be characterized more precisely.

Substituting Eq. 4 into $\epsilon_W(\bar{p}) = \rho^+(\bar{p}) - W(\bar{p})$ gives the exact closed form

$$\begin{aligned}\epsilon_W(\bar{p}) = \frac{1}{1 + z^2/s(\bar{p})}\Bigg[&\frac{z^2}{s(\bar{p})}\left(\rho^+(\bar{p}) - \frac{1}{2}\right) \\ &+ z\left(\frac{\rho^+(\bar{p})(1-\rho^+(\bar{p}))}{s(\bar{p})} + \frac{z^2}{4s(\bar{p})^2}\right)^{1/2}\Bigg].\end{aligned} \tag{29}$$

For a fixed $0 < \rho^+(\bar{p}) < 1$ and increasing $s(\bar{p})$,

$$\begin{aligned}&\left[\frac{\rho^+(\bar{p})(1-\rho^+(\bar{p}))}{s(\bar{p})} + \frac{z^2}{4s(\bar{p})^2}\right]^{1/2} \\ &\quad = \left[\frac{\rho^+(\bar{p})(1-\rho^+(\bar{p}))}{s(\bar{p})}\right]^{1/2} + O\Big(s(\bar{p})^{-3/2}\Big).\end{aligned} \tag{30}$$

Substituting Eq. 30 into Eq. 29, together with

$$\left(1 + \frac{z^2}{s(\bar{p})}\right)^{-1} = 1 + O\big(s(\bar{p})^{-1}\big),$$

gives

$$\epsilon_W(\bar{p}) = z\sqrt{\frac{\rho^+(\bar{p})(1-\rho^+(\bar{p}))}{s(\bar{p})}} + O\big(s(\bar{p})^{-1}\big). \tag{31}$$

At the all-supporting boundary $\rho^+(\bar{p}) = 1$, Eq. 4 reduces to

$$W(\bar{p}) = \frac{s(\bar{p})}{s(\bar{p}) + z^2}, \qquad \epsilon_W(\bar{p}) = 1 - W(\bar{p}) = \frac{z^2}{s(\bar{p}) + z^2}. \tag{32}$$

Thus, for fixed $0 < \rho^+(\bar{p}) < 1$, the leading correction is $O(s(\bar{p})^{-1/2})$, whereas at the all-supporting boundary it decreases as $O(s(\bar{p})^{-1})$. The $O(s(\bar{p})^{-1/2})$ statement in Theorem 3.1 is therefore a uniform bound rather than a claim that every fixed supporting-rate regime has the same asymptotic behavior.

Interpretation and assumptions. Theorem 3.1 establishes an explicit finite-evidence correction relative to the empirical supporting rate, while Corollary D.1 gives its direct promotion interpretation: under a fixed threshold, records supported by less evidence require stronger empirical support.

The identities and bounds above are algebraic properties of the Wilson score and do not themselves require independent observations. The nominal one-sided confidence interpretation additionally relies on the corresponding binomial sampling assumption. Wilson coverage is an approximation under that model, not an exact finite-sample coverage guarantee. Episode-level reduction prevents repeated annotations within one episode from directly multiplying the promotion evidence, but does not establish independence or identical distribution across episodes.

D.2 Stability of Episode-Level Evidence Aggregation

The preceding analysis characterizes how Wilson promotion treats a fixed pair of aggregated signed counts. We next consider the stability of the procedure that constructs those counts.

Proposition D.2 (Order and repetition invariance of episode-reduced aggregation)**.** *Fix the current set of local records, their abstract keys, and the deterministic pre-promotion checks. Apply the episode reduction in Appendix B.2.4 separately to each record to obtain $n_i^+(p)$, $n_i^-(p)$, and $n_i^0(p)$, and then aggregate these counts by exact abstract key as in Eq. 3. The resulting (n^+, n^-, n^0), Wilson score, and promotion decision are invariant to the processing order of this fixed input. Repeating an annotation already present for the same record and episode does not change its episode-reduced contribution.*

Proof. For a fixed record and episode, the relation-aware reducer depends on the set of support, contradiction, and no-progress annotations, not on their processing order. Repeating an annotation that is already present therefore does not change the reduced signed count or its diagnostic stall marker. The subsequent aggregation in Eq. 3 is a sum over the fixed episode-reduced counts associated with each exact abstract key, and is therefore associative and commutative. Since the abstract key and the pre-promotion checks are also fixed, the Wilson score and promotion decision are unchanged. □

Scope. Proposition D.2 concerns a fixed set of current local records. It does not make stale or out-of-order replacement safe in an asynchronous implementation; a streaming realization must separately ensure that the retained source state is current. The evaluated snapshot-rebuild specialization is described in Appendix B.3.3. Different intermediate snapshots may affect subsequent trajectories, so order-dependent acquisition does not constitute fixed input evidence.

Adding previously absent contradictory evidence or a no-progress marker can legitimately change an episode-reduced count. The repetition guarantee does not apply to occurrence-level retrieval counters. Finally, neither exact-key aggregation nor episode reduction establishes independence or identical distribution of the aggregated evidence; the nominal statistical interpretation of the Wilson bound therefore remains subject to the assumptions stated in Appendix D.

E Memory Types and Local–Global Examples

This section complements the implementation details with a unified account of what each memory component stores, what can be shared, and how the resulting records are used. The four pattern types belong to $\mathcal{P}_i^l$; together with G_i^l, $\mathcal{E}_i^l$, and $\mathcal{R}_i^l$, they constitute the local memory in Eq. 2. Global memory stores promoted abstract records from the shareable components, not a duplicate of the entire local memory.

E.1 Type Definitions and Local–Global Correspondence

Table 13 covers every component and pattern subtype. The examples are schematic adaptations of the household and PDDL examples used in this appendix, not new benchmark rollouts or literal serialized dumps. Role-level examples assume that a concrete entity has a supported task-role mapping; operational conditions are retained. "Shareable" in the sharing column means eligible for consideration, not automatically admitted to global memory. Concrete PDDL memory snapshots are shown separately in Appendix J.

Component / type	Definition	Local example	Possible global counterpart	Sharing
Graph G_i^l	Paper-level state–action–feedback transition structure with transition statistics; the serialized graph stores feedback on the corresponding state–action–next-state transition, while completed episode identifiers are retained at store level.	Soap is dirty at the basin $\rightarrow$ clean soap with basin $\rightarrow$ progress $\rightarrow$ soap is clean.	No shared graph. Separately extracted patterns or actionable relations may be considered.	Local only.
Workflow in $\mathcal{P}_i^l$	Ordered procedure spanning distinct progress phases, with conditions and an outcome.	Find a basin; clean the soap; place the clean soap in the vanity.	For a cleanable target: find water source; clean target; place in goal destination. Outcome: stored clean target.	Shareable
Precondition in $\mathcal{P}_i^l$	A temporal transition whose observed condition and action are associated with successful progress.	While carrying dirty soap at a usable basin, cleaning the soap advances the cleaning phase.	With a dirty target held at a usable cleaning tool, cleaning advances the same phase.	Shareable
Repair in $\mathcal{P}_i^l$	Successful action that resolves a failure identified in the preceding observation.	After an `invalid_action` observation, `check valid actions` resolves the recorded failure.	After `invalid_action`, `check valid actions` restores progress. This instance-free record is unchanged by abstraction.	Shareable
Failure in $\mathcal{P}_i^l$	Conditioned action–failure relation, usable to avoid repeating a failed action.	In a recorded local state, `pickup b1` leads to `invalid_action`.	An eligible anti-pattern can share a conditioned no-progress relation. The raw failure example itself remains local unless it meets the type-specific eligibility checks.	Shareable
Actionable knowledge $\mathcal{E}_i^l$	State–action–outcome relation that provides usable action or object–tool knowledge.	In the applicable soap/basin state, `clean soap with basin` produces clean soap.	Under the retained conditions, a compatible cleaning tool can clean a cleanable target. Bind both roles before use.	Shareable
Cases / reflections $\mathcal{R}_i^l$	Concrete episode examples, failure diagnoses, and recovery context.	The observation, failed `pickup b1`, diagnosis, and subsequent recovery in that episode.	No shared copy of the case or reflection. A separately extracted repair/failure relation is a different record.	Local only.

Table 13: Memory definitions, schematic examples, and sharing scope. The global column shows possible abstract records, not guaranteed promotions. Every shared record remains subject to type-specific eligibility, aggregated evidence, and the Wilson threshold. Applicability conditions and current action binding are still required when a promoted record is used.

E.2 From Local Records to Promoted Records

The global column in Table 13 describes possible abstract counterparts, not automatic promotions or interchangeable copies of local records. In particular, a supported failure pattern remains a conditioned failure relation rather than becoming a successful workflow. Table 4 gives the field-level abstraction; Tables 6 and 7 illustrate exact-key matching. Evidence aggregation and admission are specified once in Algorithm 2. Every admitted record must still be compatible with the receiving agent's current state and action bindings.

E.3 Memory Source, Record Type, and Support Role

Three distinctions are important. *Source* specifies whether an item comes from $\mathcal{M}_i^l$ or $\mathcal{M}^g$. *Record type* specifies the stored content, such as a workflow, repair, or actionable relation. *Support role* specifies its query-conditioned use in $\mathcal{H} = \{\text{ground}, \text{workflow}, \text{reflection}\}$. These are not three alternative classifications of the memory banks.

In Table 8, a local actionable record supplies grounding and a global workflow supplies action order. A local workflow can also support procedural guidance, while an applicable global record can support grounding after role binding. Local or promoted repair/failure records may supply reflection support when their conditions are triggered; this does not require sharing the concrete cases in $\mathcal{R}_i^l$. Selection and use remain governed by the current state, task phase, goal, and admissible actions, rather than by a fixed one-to-one assignment from local/global scope to a support role. This separates reusable experience from the local context needed to execute it without treating all cross-agent experience as directly applicable.

F Benchmark Domain Construction and Evaluation Scope

F.1 ALFWorld Multi-Domain Environment

To evaluate collaboration across environments, ALFWorld tasks are organized into four functional scene domains: Kitchen, Living Room, Bedroom, and Bathroom. We sample 100 training tasks from each domain for experience acquisition. Following the official protocol, evaluation uses both the *Familiar Layout* (`valid_seen`) and *Held-out Layout* (`valid_unseen`) splits. The resulting training/Familiar/Held-out counts are Kitchen $(100/20/20)$, Living Room $(100/17/11)$, Bedroom $(100/20/20)$, and Bathroom $(100/20/19)$.

Following the problem formulation, an *unseen environment* is defined with respect to the evaluated agent: it is an environment in which that agent did not acquire experience before evaluation. This agent-specific notion and the official held-out-layout split provide complementary axes of generalization. An environment may be unseen to one agent while experience from other agents is available through global memory; the Held-out Layout split additionally changes layout familiarity.

F.2 PDDL Planning and Domain Partitioning

PDDL tasks are organized into the four planning domains Gripper, Barman, Blockworld, and Tyreworld. The source sets contain 20, 20, 10, and 10 tasks, respectively, and evaluation uses a unified test set of 60 tasks. Local memory is constructed independently for each source domain before cross-agent global memory is formed.

F.3 FEVER Domain Partitioning

For FEVER, we construct two semantic source domains, Film/TV and Music, with 100 training examples in each domain. As noted in the AI Use Statement, GPT-4o-mini is used only to assist the semantic domain labeling used for this partition construction; it is not an evaluation-time memory component. Evaluation uses 60 disjoint examples, 30 from each domain, with a balanced distribution over SUPPORTS, REFUTES, and NEI labels. These partitions provide distinct sources of interaction experience while keeping the verification task and action interface unchanged.

F.4 ScienceWorld Task-Family Partitioning

ScienceWorld tasks are organized by their official task families while preserving the official train/test variation split. We sample up to 10 training variations per task template for local experience acquisition and up to 2 test variations for evaluation, yielding 60 held-out evaluation variations in the merged test set. Each local domain is evaluated with the same local-plus-global memory protocol.

F.5 Acquisition, Evaluation, and Controlled Comparisons

During acquisition, agents interact with their designated source environments, update their own local memories, and contribute selected shareable records to global promotion. Global memory is constructed only from the completed local stores. During evaluation, local and global memories are frozen: the evaluated agent refreshes its query and retrieved support at every decision step, but the test trajectory is not written back into either memory.

For each source-domain local memory, evaluation is performed on the common merged test set together with the shared global memory. Consequently, the same evaluated agent encounters both tasks aligned with its own acquisition domain and tasks from environments in which it did not acquire local experience. This operationalizes the agent-specific unseen-environment setting in the problem formulation while allowing the shared global memory to supply experience contributed by other agents. In ALFWorld, Familiar and Held-out additionally measure whether the layout itself was seen in the benchmark training split.

The controlled ablations in Appendix C reuse the same frozen local-memory snapshot. Threshold sensitivity reconstructs only the global memory from the same evidence, while the other component studies change the evaluated memory scope, promotion rule, retrieval, or composition policy without recollecting source experience.

G MemCo Per-Step Prompt Template

The displayed `Local memory`, `Global memory`, and `Failure memory` lines are rendering labels for the grounding, workflow, and reflection fields of Eq. 10, respectively; they do not define three independently stored memory banks. Record provenance is retained during composition, and an applicable global record may also contribute to grounding after current-state role binding (Appendix E.3). The displayed prompt also contains reference-case, trajectory, and insight slots outside the Memory Decision Summary, but the formatter emits an optional slot only when its list is nonempty. In the implemented `MemCo` payload, `reference_cases`, `insights`, `action_constraints`, and `repair_hints` are empty. The `MemCo` summary is supplied as at most one `planner_notes` entry after query-guided composition and rendering. `execution_patterns` is an optional wrapper slot: it is empty for ALFWorld, PDDL, and FEVER, while in ScienceWorld it may contain at most two score-improving fragments retrieved from committed local histories. Separately, the task wrapper supplies benchmark-authored few-shot reference cases from the task configuration and currently truncates that list to its first case. These surrounding wrapper slots are outside the three-field Memory Decision Summary in Eq. 10.

MemCo Per-Step Prompt Template

System Prompt

```
[ENVIRONMENT-SPECIFIC SYSTEM PROMPT]
For example, the system prompt specifies the legal action space and output format
for the current benchmark: ALFWorld household actions, PDDL operators, FEVER
Search/Lookup/Finish actions, or ScienceWorld valid commands.
```

User Prompt

```
## Successful Examples (Reference Cases; optional, task-configured)
Below are some examples of similar tasks that were successfully completed. Please
use these as references to guide your thinking and approach to the current task:
[FEW-SHOT REFERENCE CASES]
```

```
## Your Own Past Successes (Execution Patterns)
Here are examples of successful execution processes you've previously used on
similar tasks. Pay special attention to the step-by-step procedures and strategies,
especially when encountering obstacles:
[RETRIEVED SUCCESSFUL TRAJECTORIES]


## Key Insights from Related Tasks (empty in the MemCo payload)
The following are insights gathered during the execution of similar tasks. You may
refer to them during your task execution to improve problem-solving accuracy.
[RETRIEVED RULES / INSIGHTS]
[PLANNER NOTES: AT MOST ONE MemCo SUMMARY]
[ACTION CONSTRAINTS: EMPTY FOR MemCo ]
[REPAIR HINTS: EMPTY FOR MemCo ]


## How to Use Your Intermediate Findings
[ENVIRONMENT-SPECIFIC MEMORY-USAGE INSTRUCTIONS]
For example, treat retrieved memories as planning hints rather than exact action
sources; prefer currently valid actions; avoid copying stale object names, operator
arguments, evidence labels, or unavailable commands; and do not repeat actions that
failed or made no progress.


### MemCo Memory Decision Summary
Current phase: [PHASE_LABEL]
Current state: [STATE_SUMMARY]
Local memory: [LOCAL_MEMORY]
Global memory: [GLOBAL_MEMORY]
Failure memory: [FAILURE_MEMORY]
Next priority: [NEXT_PRIORITY]
Confidence / caveat: [CONFIDENCE_CAVEAT]


## Your Turn: Take Action!
Use the above examples and insights as a foundation, and now work on the following
task:
[TASK DESCRIPTION / QUERY]


[CURRENT OBSERVATION]
[CURRENT VALID ACTIONS, IF PROVIDED]
[RECENT ACTION-OBSERVATION HISTORY]


Output exactly one valid next action.
```

H Cross-Setting Statistical Significance Analysis

We conduct a complementary non-parametric analysis to assess whether the performance improvements of MemCo are consistent across backbone and task settings. Following standard practice for comparing multiple learning methods across matched evaluation settings Demšar (2006), each backbone–task configuration is treated as one evaluation block. The inputs are the method-wise mean results in Table 1. Table 14 summarizes relative standing, Table 15 tests the across-method omnibus null, and Table 16 reports the paired comparisons against each baseline. These tables answer different questions and are not additional independent experimental runs.

H.1 Matched Evaluation Blocks and Average Ranks

Evaluation blocks. The analysis contains $N_{\mathrm{blk}} = 12$ matched configurations, corresponding to three LLM backbones and four task settings: FEVER, PDDL, ALFWorld Familiar Layout, and

ALFWorld Held-out Layout. Six methods are compared in every block: ICL, A-Mem, MemSkill, MemRL, G-Memory, and MemCo, giving $K_{\mathrm{meth}} = 6$.

Accuracy and interaction steps are analyzed separately. This avoids treating the two metrics obtained from the same evaluation as independent blocks. For accuracy, larger values indicate better performance; for interaction steps, smaller values indicate better performance. Within each block, the best method receives rank 1, and tied methods receive their average rank.

Average Ranks. Let $\mathrm{rank}_{ij}^{(\mu)}$ denote the rank of method j in block i for metric $\mu \in \{\mathrm{Acc}, \mathrm{Steps}\}$. The average rank is

$$\overline{R}_j^{(\mu)} = \frac{1}{N_{\mathrm{blk}}} \sum_{i=1}^{N_{\mathrm{blk}}} \mathrm{rank}_{ij}^{(\mu)} . \tag{33}$$

Table 14 reports the resulting average ranks. MemCo obtains the best average rank for both accuracy and interaction steps. We additionally report the average over all 24 metric-specific slices as a descriptive summary; this combined value is not used as the sample size of the inferential tests. In particular, the combined column is the arithmetic mean of the two metric-specific average ranks; it is not a new set of 24 independent evaluation blocks.

Table 14: Average ranks across the matched backbone–task configurations. Lower values are better. The final column is descriptive only; statistical tests are conducted separately for accuracy and steps.

Method	Accuracy	Steps	Combined
ICL	3.750	4.333	4.042
A-Mem	5.333	3.833	4.583
MemSkill	3.583	4.000	3.792
MemRL	4.125	4.000	4.063
G-Memory	2.833	3.083	2.958
MemCo	**1.375**	**1.750**	**1.563**

H.2 Omnibus Tests Across Methods

For each metric, the Friedman statistic is

$$\chi^2_{F,\mu} = \frac{12 N_{\mathrm{blk}}}{K_{\mathrm{meth}}(K_{\mathrm{meth}}+1)} \left[\sum_{j=1}^{K_{\mathrm{meth}}} \left(\overline{R}_j^{(\mu)} \right)^2 - \frac{K_{\mathrm{meth}}(K_{\mathrm{meth}}+1)^2}{4} \right]. \tag{34}$$

When tied ranks occur, we apply the standard correction

$$C_\mu = 1 - \frac{\sum_{i=1}^{N_{\mathrm{blk}}} \sum_g \left(t_{ig}^3 - t_{ig} \right)}{N_{\mathrm{blk}}(K_{\mathrm{meth}}^3 - K_{\mathrm{meth}})}, \qquad \widetilde{\chi}^2_{F,\mu} = \frac{\chi^2_{F,\mu}}{C_\mu}, \tag{35}$$

where t_{ig} denotes the number of tied methods in tie group g of block i. Accuracy contains three pairwise ties, resulting in $C_{\mathrm{Acc}} = 0.9929$. No ties occur for interaction steps, so $C_{\mathrm{Steps}} = 1$.

Because the classical Friedman statistic can be conservative, we use the Iman–Davenport correction:

$$F_{F,\mu} = \frac{(N_{\mathrm{blk}} - 1)\widetilde{\chi}^2_{F,\mu}}{N_{\mathrm{blk}}(K_{\mathrm{meth}} - 1) - \widetilde{\chi}^2_{F,\mu}}. \tag{36}$$

Under the omnibus null hypothesis, the Iman–Davenport statistic is compared with an approximate F reference distribution with $K_{\mathrm{meth}} - 1 = 5$ and $(K_{\mathrm{meth}} - 1)(N_{\mathrm{blk}} - 1) = 55$ degrees of freedom.

As reported in Table 15, the omnibus null hypothesis is rejected for both metrics under this testing model. This omnibus result indicates an across-method difference, but does not establish which individual baseline differs from MemCo; those comparisons are reported next.

Table 15: Friedman tests with the Iman–Davenport correction. The reported Friedman statistics include the tie correction where applicable.

Metric	$\widetilde{\chi}_F^2$	F_F	**df**	p
Accuracy	30.324	11.240	$(5, 55)$	1.71×10^{-7}
Steps	15.571	3.855	$(5, 55)$	4.58×10^{-3}

H.3 Paired Comparisons, Effect Sizes, and Interpretation

After rejecting the omnibus null hypotheses, we compare MemCo with each baseline using exact two-sided Wilcoxon signed-rank tests. Unlike a comparison based only on average ranks, this paired test uses the direction and magnitude of the performance differences within each matched configuration.

For baseline b, we define the accuracy difference as

$$\Delta_{i,b}^{\mathrm{Acc}} = \mathrm{Acc}_{i,\texttt{MemCo}} - \mathrm{Acc}_{i,b}, \tag{37}$$

and the interaction-step difference as

$$\Delta_{i,b}^{\mathrm{Steps}} = \mathrm{Steps}_{i,b} - \mathrm{Steps}_{i,\texttt{MemCo}}. \tag{38}$$

Thus, positive differences favor MemCo for both metrics. Zero differences are excluded from the signed-rank statistic. The remaining absolute differences are ranked, with average ranks assigned to ties. Let T_{W}^+ and T_{W}^- be the sums of ranks associated with positive and negative differences, respectively. The two-sided test statistic is

$$T_{\mathrm{W}} = \min\{T_{\mathrm{W}}^+, T_{\mathrm{W}}^-\}. \tag{39}$$

We calculate the conditional two-sided p-values by enumerating all sign assignments to the observed nonzero absolute-difference ranks under the paired sign-symmetry null. Here T_{W} denotes the signed-rank statistic; $W(\bar{p})$ remains the Wilson promotion score. The enumeration is exact for this sign-assignment model, including averaged tied ranks; it does not itself establish that the evaluation blocks are independent.

Because the analysis contains five control-versus-baseline comparisons for each of two metrics, we jointly correct all $M = 10$ post-hoc p-values using Holm's step-down procedure Holm (1979). Given the ordered raw p-values $p_{(1)} \le \cdots \le p_{(M)}$, the adjusted values are

$$\widetilde{p}_{(i)} = \max_{1 \le j \le i} \left\{ \min\left[1, (M - j + 1) p_{(j)}\right] \right\}. \tag{40}$$

This correction controls the family-wise error rate across all reported method–metric comparisons.

Effect Size. In addition to statistical significance, we report the matched-pairs rank-biserial correlation

$$r_{\mathrm{rb}} = \frac{T_{\mathrm{W}}^+ - T_{\mathrm{W}}^-}{T_{\mathrm{W}}^+ + T_{\mathrm{W}}^-}. \tag{41}$$

Positive values indicate an advantage for MemCo, with values closer to 1 indicating greater cross-configuration consistency.

Table 16 reports the complete results. The Win/Loss/Tie column counts the configurations in which MemCo is better, worse, or tied with the corresponding baseline. For example, the G-Memory comparisons give 10 wins and 2 losses for each metric, with positive rank-biserial effects and Holm-adjusted $p = 0.0146$. The win counts summarize direction, whereas the effect sizes and signed-rank statistics also depend on the ranked magnitudes of the paired differences.

Table 16: Exact two-sided Wilcoxon signed-rank comparisons between MemCo and each baseline across the 12 matched backbone–task configurations. Holm correction is applied jointly to all ten comparisons. All differences are defined such that positive values favor MemCo.

Metric	Baseline	Win/Loss/Tie	T_{W}	r_{rb}	Exact p	Holm-adjusted p
Accuracy	ICL	11/1/0	1	0.974	9.77×10^{-4}	8.79×10^{-3}
	A-Mem	12/0/0	0	1.000	4.88×10^{-4}	4.88×10^{-3}
	MemSkill	11/1/0	2	0.949	1.46×10^{-3}	1.03×10^{-2}
	MemRL	11/0/1	0	1.000	9.77×10^{-4}	8.79×10^{-3}
	G-Memory	10/2/0	6	0.846	6.84×10^{-3}	1.46×10^{-2}
Steps	ICL	10/2/0	4	0.897	3.42×10^{-3}	1.46×10^{-2}
	A-Mem	11/1/0	3	0.923	2.44×10^{-3}	1.46×10^{-2}
	MemSkill	11/1/0	3	0.923	2.44×10^{-3}	1.46×10^{-2}
	MemRL	9/3/0	6	0.846	6.84×10^{-3}	1.46×10^{-2}
	G-Memory	10/2/0	5	0.872	4.88×10^{-3}	1.46×10^{-2}

Summary. Under the stated block-level testing assumptions, all ten paired comparisons have Holm-adjusted $p < 0.05$. In particular, the improvements over G-Memory remain significant for both accuracy ($p_{\mathrm{Holm}} = 0.0146$) and interaction steps ($p_{\mathrm{Holm}} = 0.0146$). The corresponding rank-biserial correlations are 0.846 and 0.872, respectively, indicating that the differences are not only statistically significant but also directionally consistent across the evaluated configurations.

The unit of analysis in these tests is the matched backbone–task configuration. The two metrics are tested separately, and the analysis does not treat the three repeated runs or the two metrics from one configuration as additional independent evaluation blocks. Accordingly, these results should be interpreted as evidence of consistent performance differences across the evaluated configurations rather than as a claim about all possible models and environments. Configurations reuse backbones and task families, so dependence between blocks may limit that inferential interpretation; the reported p-values are conditional on the block-level assumptions and are not a substitute for independent task- or seed-level replication. The average ranks and paired differences remain descriptive summaries of the evaluated configurations.

I Case Study: Adaptive Memory Support and Cross-Environment Generalization

To qualitatively evaluate MemCo, we analyze the trajectories across heterogeneous settings to investigate how local situational grounding and global procedural workflows interact dynamically online.

Exploration Efficiency. As shown in Fig. A.1(a) and A.1(b), when facing the same task "Put a clean cloth in countertop", the displayed G-Memory trajectory attempts to place the cloth before satisfying the cleanliness requirement, resulting in redundant interactions. In contrast, MemCo retrieves decision-relevant support before placement: the reusable clean-before-store ordering is combined with local actionable knowledge for the current state. The resulting trajectory avoids the invalid move and completes the task with fewer interaction steps.

Unseen-Environment Generalization. The comparison between Fig. A.1(b) and A.1(c) illustrates how shared experience supports an agent in an unseen environment. The task changes from cleaning a cloth in a kitchen to "Clean a bar of soap and put it in a vanity" in a bathroom. A kitchen-specific instruction can be irrelevant in the bathroom, whereas the clean-before-place ordering can remain useful under the corresponding applicability conditions.

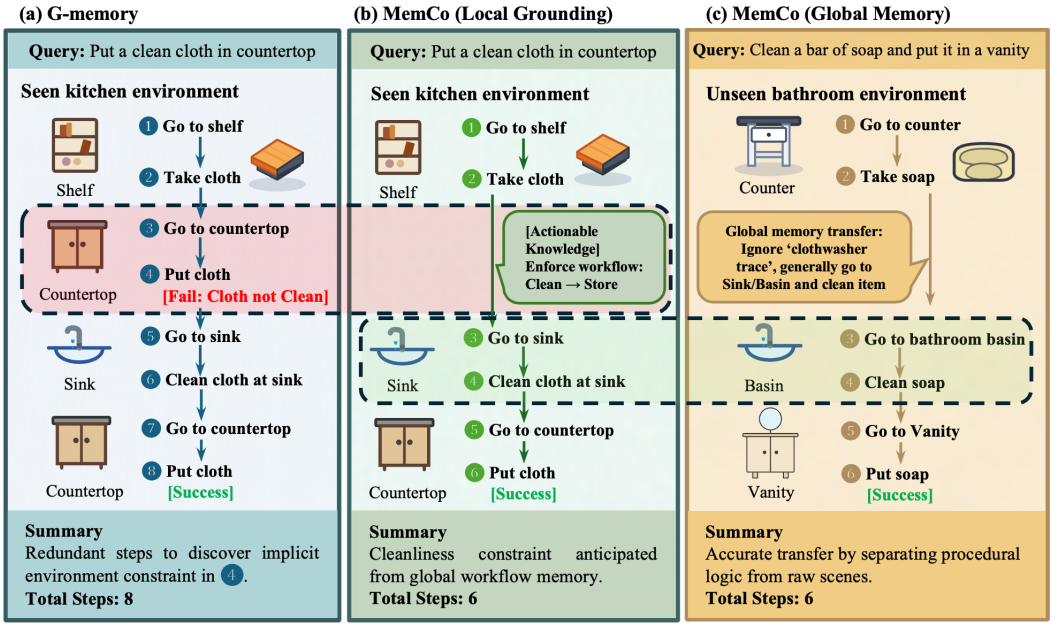


Figure A.1: **Case study showcasing adaptive memory support and cross-environment generalization.** (a) In the shown G-Memory trajectory, the agent attempts placement before satisfying the cleanliness requirement. (b) MemCo uses local actionable knowledge to avoid the invalid placement in the same environment. (c) In an unseen environment, MemCo combines a reusable global workflow with current-environment grounding.

As illustrated in Fig. A.1(c), query-guided selection and composition exclude incompatible kitchen-specific guidance, including the irrelevant "clothwasher trace". The shared workflow supplies the action order; the current observation and available actions ground its roles in the bathroom basin and vanity, with applicable local evidence providing further context. This decision-time grounding does not require writing new environment-specific records into the frozen evaluation memory. The shown trajectory completes the task in 6 steps, illustrating how shared procedural guidance can be transferred to an environment unseen by the evaluated agent while current-environment grounding determines how that guidance is executed.

J Visualizations of PDDL Memory Structures

To present a clear and concrete view of our memory repository, we visualize the actual global and local memory structures generated during the PDDL symbolic planning tasks. Because these structures are very wide, each visualization and its caption are rotated together and placed on a standalone figure page to use the page height and improve readability.

J.1 Global Memory Structure

Figure A.2 shows the global memory snapshot assembled from contributions in gripper, blockworld, barman, and tyreworld. The sharing interface in Algorithm 2 receives selected reusable records and their evidence rather than complete local trajectories or graphs. The visualization contains promoted workflows and precondition relations from different planning domains, illustrating how recurrent local structures are represented in the shared global bank. The stored global records remain conditional and are checked against the current planning state before use.



Figure A.2: **Visualization of the final PDDL global memory structure.** The centralized server aggregates top-promoted rules and preconditions across heterogeneous planning domains into a structured global bank.

J.2 Local Memory Graphs

Figures A.3 through A.6 display local-memory snapshots across the four PDDL domains, including state and interaction structure, outcome-annotated records, and sample artifacts. The connected experience and concrete cases remain local; selected patterns and actionable relations may be shared through the separate sharing interface.

Figure A.6 provides examples of the stored local relations, including goal-advancing conditions, failure-triggered repairs, and failure-avoidance guidance. These entries illustrate the complementary roles of structured patterns, actionable knowledge, and concrete local context. The visualization labels reflect the stored instances; the paper-level memory components remain the G–$\mathcal{P}$–$\mathcal{E}$–$\mathcal{R}$ decomposition in Eq. 2.

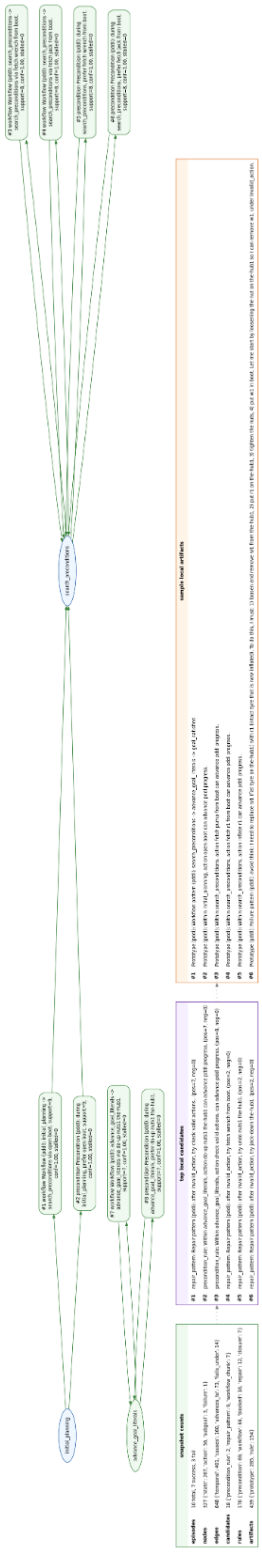


Figure A.3: **Local memory structure for the tyreworld domain.** The graph captures real-time planning states and local records for vehicle repair configurations.

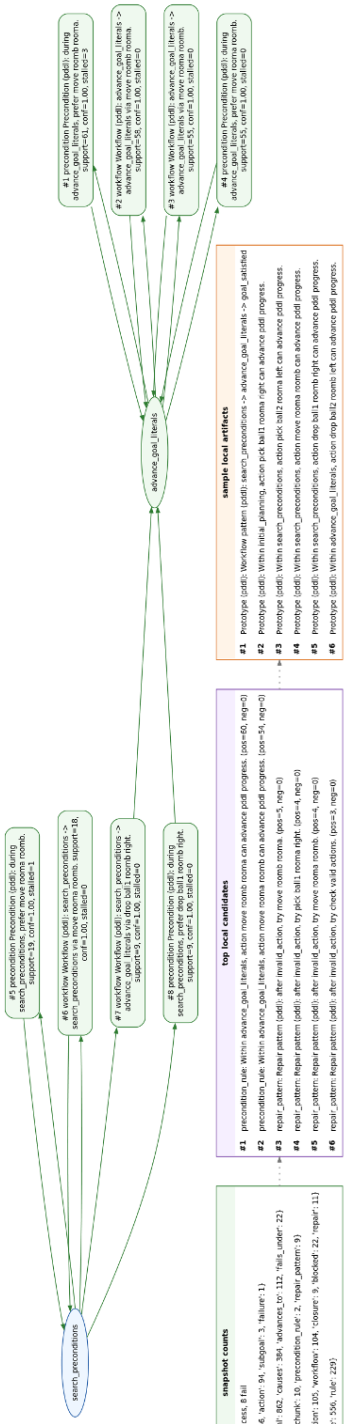


Figure A.4: **Local memory structure for the gripper domain.** The graph visualizes execution trajectories and goal-advancing action links for robotic manipulation tasks.

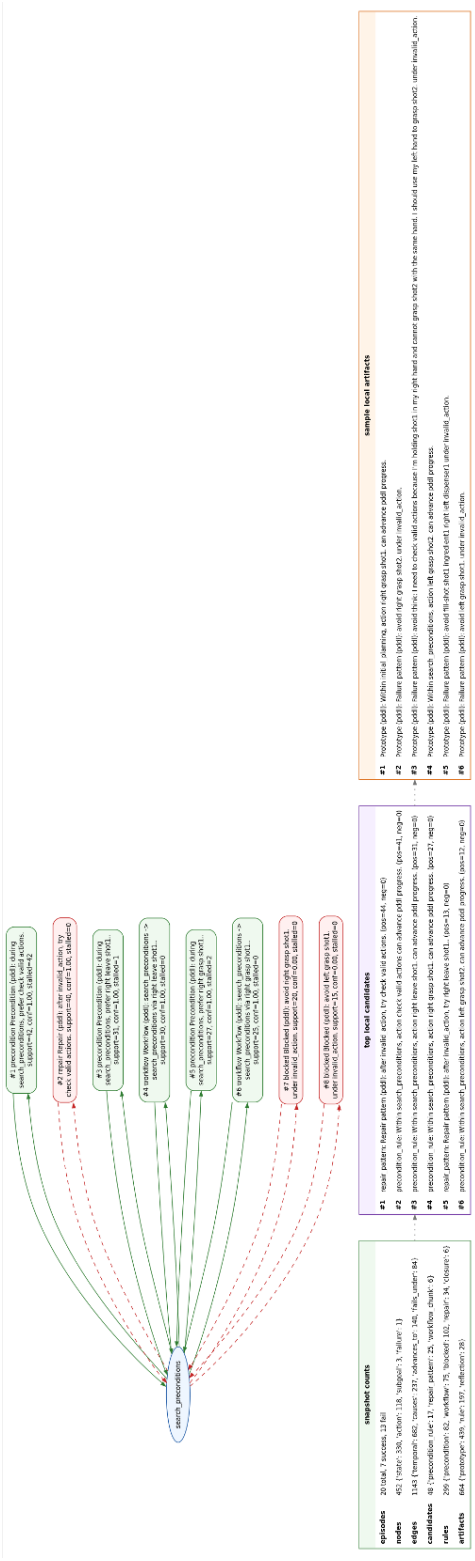


Figure A.5: **Local memory structure for the barman domain.** The snapshot logs successful action combinations and filters out invalid interaction paths in a dense action space.

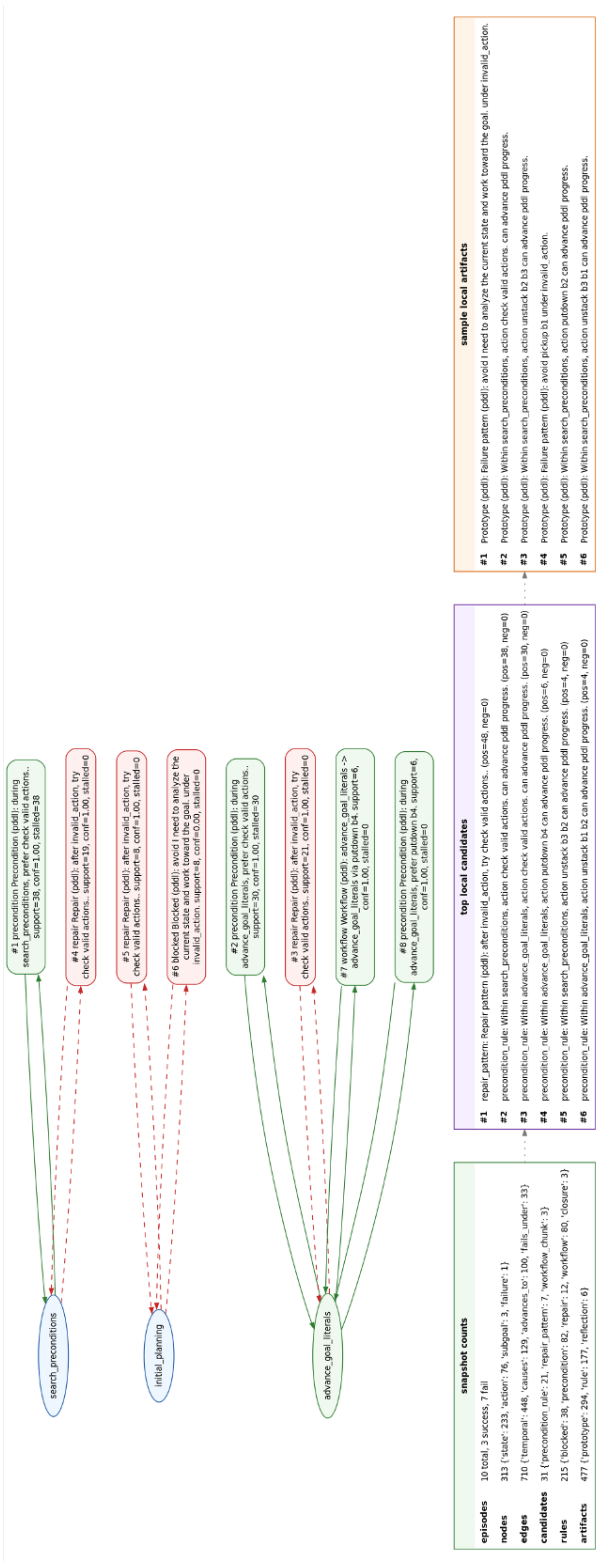


Figure A.6: **Local memory structure for the blockworld domain.** The network represents temporal sequence steps and block rearrangement heuristics.