BDI ARCHITECTURE FOR EFFICIENT ORCHESTRATION OF LLM-BASED MULTI-AGENT SYSTEMS

Anonymous authors
Paper under double-blind review

ABSTRACT

LLM-based multi-agent systems (MAS) are shifting from manually designed workflows toward automated orchestration, which often relies on repeated workflow search and optimization or additional training. We, human-beings, do not reconsider all possibilities whenever a course of action fails, but maintain an intention and revise the plan only when it is no longer appropriate for reasoning. Inspired by this phenomenon, we propose a BDI (Belief–Desire–Intention)-based orchestration framework that dynamically constructs and adapts MAS workflows at inference time without explicit workflow search or additional training. A single orchestrator maintains beliefs about agent states and execution outcomes, represents candidate sub-goals as desires, and commits to a selected sub-goal as an intention. It then constructs and adapts an execution plan based on the current system state and accumulated experience. When plan adaptation is insufficient to achieve the intended objective, the orchestrator abandons its current intention and selects a new sub-goal from its desires. This intention commitment mechanism reduces repeated deliberation while preserving adaptability to changing execution conditions. Experiments across agentic and general-domain reasoning benchmarks show that our framework improves GAIA accuracy by 4.24 percentage points and ALFWorld success rate by 19.40%p over the strongest baselines, while achieving the highest general-domain average of 80.00 with a GAIA execution cost of $0.0127 per task.

1 INTRODUCTION

LLM-based multi-agent systems (MAS) have shown strong potential for solving complex tasks by coordinating multiple specialized agents (Wu et al., 2024; Hong et al., 2024; Li et al., 2023; Chen et al., 2024). Recent work increasingly moves beyond manually designed workflows toward automated orchestration, using workflow search and optimization (Zhang et al., 2025b;a; Wang et al., 2025) or learning orchestration strategies through additional training (Ye et al., 2025; Gao et al., 2026). However, a key challenge in automated orchestration is how much of the current decision process should be reconsidered when execution does not proceed as expected. As illustrated in Fig. 1(a), one approach is to re-plan after every execution step, potentially invoking workflow search or optimization repeatedly (Zhang et al., 2025b;a; Wang et al., 2025). This allows the workflow to incorporate newly observed outcomes and recover from intermediate errors, but requires repeated planner calls and incurs substantial orchestration overhead. Another approach is to construct the entire workflow once and execute it without revision (Fig. 1(b)). While this avoids repeated planning, an erroneous intermediate result may remain uncorrected and propagate through subsequent steps to the final answer.

We take a different view of orchestration. Humans rarely reconsider all possible objectives whenever a course of action fails. Instead, they tend to remain committed to the objective they are currently pursuing and revise how to achieve it based on newly observed outcomes. Only when such adjustments no longer provide a viable path forward do they reconsider what should be pursued. This pattern closely follows the Belief-Desire-Intention (BDI) model (Bratman, 1987; Rao & Georgeff, 1995), which distinguishes the current state of the world as beliefs, possible objectives as desires, and the objective currently being pursued as an intention. We adopt this distinction to separate adaptation of how to achieve the current objective from reconsideration of the objective itself.

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.

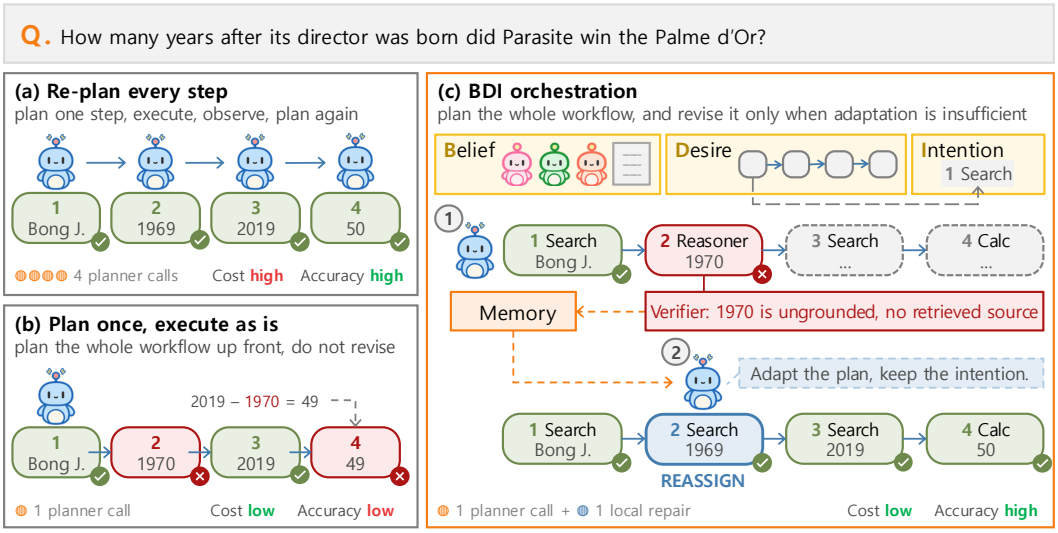

Figure 1: **Comparison of orchestration strategies.** (a) Step-wise re-planning adapts to intermediate outcomes but incurs repeated planning overhead. (b) One-shot planning is efficient but cannot recover from erroneous intermediate results. (c) Our BDI-based orchestration retains the current intention and locally adapts the execution plan, achieving adaptive reasoning with reduced orchestration cost.

Motivated by this, we propose a BDI-based orchestration framework that dynamically constructs and adapts MAS workflows at inference time without explicit workflow search or additional training. A single orchestrator maintains beliefs over agent states, intermediate outputs, and execution outcomes, represents candidate sub-goals as desires, and commits to one desire as the current intention. It then constructs an execution plan by selecting and coordinating agents and continuously updates its beliefs as execution proceeds. The updated beliefs guide plan adaptation while the current intention remains fixed. As illustrated in Fig. 1(c), when verification identifies an unreliable intermediate result, the orchestrator revises the plan by reassigning the failed reasoning step to a search agent and continues toward the same intention, rather than reconstructing the entire workflow. Only when plan adaptation is insufficient does it revise the intention and select a new sub-goal from the remaining desires. This intention commitment mechanism reduces unnecessary deliberation while preserving adaptability to changing execution conditions. The key contributions of this paper are summarized as follows.

- **BDI-based dynamic orchestration**: We formulate dynamic MAS orchestration in terms of beliefs, desires, and intentions, enabling workflows to be constructed and adapted at inference time without explicit workflow search or additional training.
- **Intention commitment for adaptive execution**: We propose an intention commitment mechanism that maintains the current intention while adapting its execution plan, and revises the intention only when plan-level adaptation is insufficient. This separates how to achieve the current objective from which objective to pursue.
- **Effective and efficient multi-agent reasoning**: Across diverse agentic and general-domain reasoning benchmarks, our framework achieves strong overall performance while maintaining efficient inference through selective adaptation rather than repeated global re-planning.

2 Related Work

Orchestration in LLM multi-agent systems. Early LLM-based multi-agent systems rely on manually designed agent roles, execution orders, and interaction patterns (Wu et al., 2024; Hong et al., 2024; Li et al., 2023; Chen et al., 2024). Recent work has increasingly automated this process

through workflow search and optimization (Zhang et al., 2025b;a; Wang et al., 2025) or by training models to generate orchestration strategies (Ye et al., 2025; Gao et al., 2026). While these approaches reduce manual workflow design, they introduce additional search or training cost and largely determine the orchestration strategy before execution. Other approaches adapt orchestration during execution. Magentic-One (Fourney et al., 2024), OWL (Hu et al., 2025), MAS^2 (Wang et al., 2026), and Agent-Co-op (Shen et al., 2026) revise task plans, system configurations, or agent interactions based on execution feedback. Such runtime adaptation improves robustness to changing execution conditions, but can require repeated replanning or learned rectification mechanisms. More importantly, when a local execution failure occurs, reconsidering a broader portion of the coordination process than necessary can introduce substantial orchestration overhead and discard useful progress already made. Our framework instead emphasizes commitment during adaptation. Drawing on the BDI model (Bratman, 1987; Rao & Georgeff, 1995), the orchestrator continuously updates its beliefs while maintaining the current intention, and first adapts the execution plan using the information accumulated so far. Verified intermediate results are preserved, and only the affected part of the plan is revised when possible. The intention is reconsidered only when plan-level adaptation is insufficient. By limiting re-deliberation to the level required by the observed outcome, our framework enables dynamic orchestration at inference time without explicit workflow search or additional training.

Memory in LLM agents. Memory enables LLM agents to leverage past information and experience (Zhang et al., 2025c). Existing approaches represent memory in textual or structured forms (Packer et al., 2024; Chhikara et al., 2025) and retrieve relevant information to improve subsequent reasoning and action generation. Reflexion (Shinn et al., 2023) stores verbal reflections from failed attempts, A-MEM (Xu et al., 2025) organizes and evolves interconnected memories, Memp (Fang et al., 2026) distills past trajectories into reusable procedural knowledge, and ReasoningBank (Ouyang et al., 2026) extracts strategy-level memories from successful and failed experiences. In these methods, retrieved memories primarily provide context or guidance to the acting agent, while the model determines how to use them during reasoning. Consequently, retrieving more experience does not necessarily improve performance, and irrelevant or excessive memories may introduce noise; ReasoningBank, for example, reports performance degradation as more experiences are retrieved (Ouyang et al., 2026). More importantly for multi-agent orchestration, existing memory mechanisms mainly support how an agent should act, rather than determining which intermediate results should be preserved and which part of an ongoing plan should be revised in response to execution outcomes. In our framework, beliefs are used not merely to reuse past experience, but to maintain the execution state for adaptive orchestration. This allows the orchestrator to preserve valid progress while revising only what is necessary, extending the role of memory from guiding agent behavior to determining the scope of orchestration adaptation.

3 METHOD

3.1 PROBLEM DEFINITION

Given a query q, a registry of available agent capabilities A, and a set of tools $\mathcal{T}$, the orchestrator must construct and execute a workflow that produces an answer y under an output constraint ϕ.

At execution step t, the orchestrator maintains

$$B_t = (E_t, H_t), \qquad D_t = \{g_t^{(1)}, \ldots, g_t^{(m)}\}, \qquad I_t = g_t^*, \qquad P_t = (V_t, R_t), \tag{1}$$

where B_t denotes the Belief state, containing accumulated execution evidence and history. D_t denotes the set of candidate sub-goals, and $I_t \in D_t$ is the sub-goal currently selected as the Intention. P_t is the execution plan for achieving I_t, represented as a dependency graph over executable subtasks. The orchestration problem is to adapt P_t as the Belief evolves while preserving I_t whenever possible, and to revise I_t only when plan-level adaptation is insufficient.

3.1.1 COMMITMENT AND EXECUTION

A central problem in multi-agent orchestration is deciding what to do when an intermediate step fails. A common approach is to regenerate the entire workflow. This discards correct results, repeats

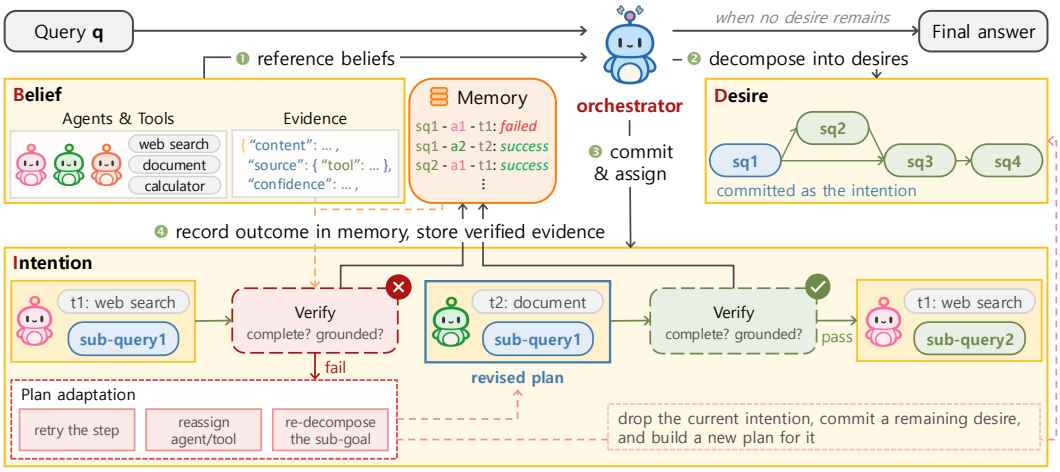

Figure 2: **Overview of our BDI-based orchestration.** The orchestrator maintains beliefs, selects a desire as the current intention, and executes a plan while verifying intermediate results. Upon failure, it first adapts the plan under the same intention while preserving verified evidence, and revises the intention only when plan-level adaptation is insufficient.

work that has already been paid for, and can make the system oscillate between plans without converging. Our orchestrator instead follows the BDI principle of *commitment*: once it has chosen what to achieve, it changes only *how* it pursues that goal, for as long as a viable way forward remains.

Committing to an Intention. Given a query, the orchestrator decomposes it into a dependency graph of sub-goals, each describing an intermediate result needed to answer the query. It commits to this graph as the current Intention I_t. The Intention specifies *what* must be achieved, while the execution plan P_t specifies *how*: which agent and tool handles each sub-goal and with what inputs. This separation allows the plan to change while the goal stays fixed.

Executing and updating the Belief. Sub-goals are executed in dependency order, and each receives the results of its predecessors as context. Every result is verified for *completeness* (does it fully address the sub-goal?), *grounding* (is it supported by the collected evidence?), and *consistency* (does it agree with what is already known?). Results that pass are written into the Belief B_{t+1}, which serves as a store of trusted progress that later steps reuse rather than recompute.

When to stay committed. We define intention commitment as

$$\Pi(I_t \mid B_{t+1}) \neq \emptyset \quad \Longrightarrow \quad I_{t+1} = I_t, \tag{2}$$

where $\Pi(I_t \mid B_{t+1})$ denotes the set of plan repairs that can still achieve I_t under the updated Belief. In other words, a failed step is not by itself a reason to abandon the goal.

3.1.2 Reconsideration

Commitment does not mean persisting blindly. The question is how much of the current course of action should change in response to a failure. Our answer is as little as possible: the orchestrator tries the smallest repair that could fix the problem, and escalates only when that repair proves insufficient.

Repair ladder. The available repairs are ordered from local to global:

- Retry: re-execute the sub-goal with feedback on what was missing or wrong.
- Augment: supply the missing information or add a supporting step.
- Reassign: hand the sub-goal to a better-suited agent or tool.
- Replan: re-decompose the sub-goal together with its neighbors in the graph.

Algorithm 1 Commitment and reconsideration in the BDI orchestrator

Require: query q, failure memory $\mathcal{M}$
1: $D \leftarrow q$; $B \leftarrow \emptyset$
2: $I \leftarrow \mathrm{Decompose}(D, B)$; $P \leftarrow \mathrm{Plan}(I, B)$ ▷ commit to an Intention
3: **while** I has an unfinished sub-goal **do**
4: $n \leftarrow$ next sub-goal of I whose dependencies are resolved
5: $r \leftarrow \mathrm{Execute}(n, P, B)$
6: **if** $\mathrm{Verify}(r, B)$ **then**
7: $B \leftarrow B \cup \{r\}$ ▷ store verified progress
8: **else**
9: $k \leftarrow \mathrm{NextRung}(n, r)$ ▷ start by failure type, then escalate
10: $k \leftarrow \max\big(k, \mathcal{M}.\mathrm{Recommend}(n, r)\big)$ ▷ memory may skip rungs
11: **if** $k < K$ **then**
12: $P \leftarrow \mathrm{Adapt}_k(P, B)$ ▷ keep I, repair locally
13: **else**
14: $I \leftarrow \mathrm{Select}(D \mid B)$; $P \leftarrow \mathrm{Plan}(I, B)$ ▷ reconsider
15: **end if**
16: **end if**
17: **end while**
18: **return** $\mathrm{Synthesize}(D, B)$

Even REPLAN is confined to the neighborhood of the failure, so sub-goals elsewhere and their verified results are left untouched.

The starting rung depends on the type of failure. A result that is on the right track but incomplete starts with RETRY, since a small correction is likely to suffice. A result that contradicts the Belief suggests that the decomposition itself is flawed, so repair starts at REPLAN. Each subsequent failure moves the orchestrator one rung up.

When to reconsider. Let k_t denote the current rung and K the number of rungs. The orchestrator decides

$$(I_{t+1}, P_{t+1}) = \begin{cases} \big(I_t,\ \mathrm{Adapt}_{k_t}(P_t, B_{t+1})\big), & \text{if } k_t < K, \\ (I',\ \mathrm{Plan}(I', B_{t+1})), & \text{otherwise,} \end{cases} \tag{3}$$

where Adapt_k applies the k-th repair and

$$I' = \mathrm{Select}(D_t \mid B_{t+1}) \tag{4}$$

selects a new Intention from the Desire given the current Belief. The first case corresponds to Eq. equation 2: a repair remains, so I_t is kept. The second case arises only after every repair has failed. The orchestrator then drops the unachievable sub-goal and recommits to the remaining sub-goals of the Desire, with the Belief carried over intact.

This design makes the orchestrator *stable*, in that a single failure neither overturns the goal nor forces verified work to be redone, and *responsive*, in that its reaction grows as failures accumulate. Algorithm 1 summarizes the procedure.

Failure memory. Climbing the ladder one rung at a time is safe but wasteful when the same kind of failure recurs. If a certain type of sub-goal is consistently mishandled by one agent, for instance, RETRY and AUGMENT will fail every time before REASSIGN is reached. The orchestrator therefore records, for each kind of failure, which repair eventually succeeded, as judged by the verifier rather than by ground-truth labels. When a similar failure recurs, repair starts from the previously successful rung (line 10 of Algorithm 1). Because memory can only raise the starting rung, it cuts wasted attempts and the tokens spent on them without altering when the Intention is reconsidered.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETTING

Benchmarks and backbone. We evaluate our BDI-based orchestration framework on two agentic benchmarks and four general-domain reasoning benchmarks. GAIA evaluates tool-augmented reasoning using 165 validation tasks, comprising 53 Level 1, 86 Level 2, and 26 Level 3 tasks. ALFWorld evaluates sequential decision making on 134 unseen episodes. To assess applicability beyond explicitly agentic environments, we also evaluate on HotpotQA, HumanEval, MBPP, and MATH, covering multi-hop question answering, code generation, and mathematical reasoning. Our experimental protocol fixes the backbone to GPT-4o-mini with temperature 0 for all LLM roles.

Baselines. We compare against four families of methods: *fixed execution* (Vanilla/CoT and Fixed MAS), *memory-based adaptation* (Reflexion, A-MEM, and Memp), *workflow or architecture optimization* (AFlow and MaAS), and *dynamic runtime orchestration* (MAS^2, OWL/Workforce, and Agent-Co-op). Our method explicitly maintains belief, desire, and intention states to determine when to retain or reconsider the current intention.

Evaluation metrics. We report overall and per-level accuracy on GAIA, and success rate (SR) and mean execution steps among successful episodes on ALFWorld. For the general-domain benchmarks, we report HotpotQA F1, HumanEval and MBPP pass@1, and MATH accuracy, all on a 0–100 scale. We summarize general-domain performance using the unweighted arithmetic mean of the four scores; methods with missing scores are excluded from this aggregate comparison. We evaluate inference-time resource usage on GAIA using tokens, LLM calls, and execution cost per task. For the ALFWorld ablations, we additionally report re-planning events and intention switches per task.

4.2 EXPERIMENTAL RESULTS

4.2.1 PERFORMANCE ON AGENTIC BENCHMARKS

Table 1 compares our framework with fixed-execution, memory-based, workflow-optimization, and runtime-orchestration baselines on GAIA and ALFWorld. On GAIA, our method achieves the highest overall accuracy of 21.82%, improving over the strongest baseline, OWL/Workforce (17.58%), by 4.24 percentage points. The improvement is mainly observed on Level 1 and Level 2, where our method reaches 32.10% and 21.20%, respectively, compared with the best baseline scores of 26.42% and 18.60%. On the more difficult Level 3 tasks, our method achieves 3.85%, matching several baselines rather than improving over them. This suggests that the gain mainly comes from more reliable orchestration on tasks for which intermediate failures can still be recovered, while the hardest cases remain challenging across methods.

On ALFWorld, our method achieves a success rate of 43.28%, substantially higher than the strongest available baseline, Agent-Co-op (23.88%). Fixed execution and memory-based methods obtain lower success rates, while OWL/Workforce achieves only 3.73% despite adapting its plan during execution. These results are consistent with our motivation that runtime adaptation benefits from preserving the current objective and revising the execution plan when failures occur, rather than either executing a fixed workflow unchanged or broadly reconsidering the workflow. However, our method does not achieve the fewest execution steps: it requires 24.00 steps on average among successful episodes, compared with 11.04 for A-MEM. The higher success rate therefore does not result from shorter trajectories, but from completing a larger fraction of episodes despite requiring additional interaction in some cases.

4.2.2 PERFORMANCE ON GENERAL-DOMAIN BENCHMARKS

Table 2 reports results on HotpotQA, HumanEval, MBPP, and MATH using the same GPT-4o-mini backbone. Our method achieves the highest average score of 80.00 among methods with complete results, outperforming the strongest baseline average, MaAS (78.31), by 1.69 percentage points. In particular, our method obtains the best performance on HotpotQA (77.56) and HumanEval (96.84). These results suggest that the proposed orchestration strategy is not limited to explicitly agentic environments and can also be applied to tasks requiring multi-hop reasoning and code generation.

Table 1: **Performance on agentic benchmarks.** GAIA reports level-wise and overall accuracy (%), while ALFWorld reports success rate (SR, %) and average execution steps for successful episodes.

	GAIA				ALFWorld	
Method	Level 1	Level 2	Level 3	Overall	Success Rate ↑	Steps ↓
Fixed Execution						
Vanilla / CoT	15.09	9.30	0.00	9.70	15.67	20.00
Fixed MAS	26.42	10.47	**3.85**	14.55	17.91	24.38
Memory-based Adaptation						
Reflexion	24.53	8.14	0.00	12.12	12.69	16.18
A-MEM	20.75	9.30	**3.85**	12.12	20.15	**11.04**
Memp	–	–	–	–	15.67	15.62
Workflow / Architecture Optimization						
AFlow	16.98	12.79	0.00	12.12	–	–
MaAS	15.09	11.63	0.00	10.91	–	–
Dynamic / Runtime Orchestration						
OWL / Workforce	22.64	18.60	**3.85**	17.58	3.73	26.20
Agent-Co-op	16.98	9.30	**3.85**	10.91	23.88	23.12
BDI-based Orchestration						
Ours	**32.10**	**21.20**	**3.85**	**21.82**	**43.28**	24.00

The improvement is not uniform across all benchmarks. On MBPP, our method achieves 76.20, below OWL/Workforce (78.40), while on MATH it reaches 69.40, compared with 75.00 for MaAS. This indicates that dynamic orchestration does not consistently outperform methods that are well suited to particular task structures. Rather, the overall results suggest that maintaining an intention while adapting its execution plan provides a reasonably general mechanism across heterogeneous reasoning tasks. Importantly, this performance is obtained without benchmark-specific workflow search or additional training, which is consistent with our goal of performing adaptive orchestration directly at inference time.

4.2.3 Inference-Time Efficiency

Table 3 compares the inference-time resource usage of different orchestration strategies on GAIA. Our method uses 98.1K tokens per task with an average cost of $0.0127, substantially lower than most fixed-execution, memory-based, and runtime-orchestration baselines. In particular, compared with OWL/Workforce, which repeatedly revises the workflow during execution, our method reduces token usage from 1198.2K to 98.1K and cost from $0.1840 to $0.0127. It also requires fewer tokens and lower cost than AFlow, despite performing adaptation directly at inference time.

Our method does not minimize every resource measure. It makes 35.70 LLM calls per task, more than most baselines and second only to OWL/Workforce. This is consistent with its step-wise execution and verification procedure, where intermediate outcomes are explicitly checked before subsequent adaptation. Nevertheless, these calls remain relatively lightweight in terms of total token usage and monetary cost. MAS^2 achieves the lowest overall cost ($0.0011), and MaAS and MAS^2 use substantially fewer tokens than our method. Thus, the results do not indicate that our framework is universally the cheapest approach; rather, they show that runtime adaptation can be performed with moderate inference cost without repeatedly reconstructing the entire workflow. Together with the performance results in Table 1, this supports our motivation for preserving the current intention and restricting adaptation to the necessary part of the execution plan.

4.2.4 Ablation Study

Table 4 examines the contribution of each BDI component and the effect of different commitment strategies on ALFWorld. Removing any of the three components degrades performance. In particular, disabling Belief updates or Desire filtering reduces the success rate from 43.28% to 11.94% and 16.42%, respectively, indicating that maintaining an up-to-date execution state and restricting

Table 2: **Performance on general-domain reasoning benchmarks using GPT-4o-mini.** Results are reported on HotpotQA, HumanEval, MBPP, and MATH, with Avg. denoting the unweighted mean across all four benchmarks.

Method	HotpotQA	HumanEval	MBPP	MATH	Avg.
Fixed Execution					
Vanilla / CoT	72.23	89.31	69.00	70.40	75.24
Fixed MAS	63.54	87.02	57.80	67.80	69.04
Memory-based Adaptation					
Reflexion	76.25	82.44	69.20	31.20	64.77
A-MEM	68.30	84.73	67.20	61.00	70.30
Memp	74.11	80.92	70.40	68.80	73.56
Workflow / Architecture Optimization					
AFlow	63.54	83.97	55.20	62.40	66.28
MaAS	75.09	93.13	70.00	**75.00**	78.31
Dynamic / Runtime Orchestration					
OWL / Workforce	73.58	83.21	78.40	62.00	74.30
Agent-Co-op	61.74	90.08	53.80	69.00	68.65
BDI-based Orchestration					
Ours	**77.56**	**96.84**	76.20	69.40	**80.00**

Table 3: **Efficiency per task on GAIA.** Tok. denotes total tokens per task (in thousands), Calls the number of LLM calls, and Cost the execution cost in USD.

Method	Tok. (K)	Calls	Cost ($)
Vanilla / CoT	399.9	1.01	0.0612
Fixed MAS	515.0	2.92	0.0793
Reflexion	473.3	2.99	0.0730
A-MEM	437.3	1.01	0.0670
Memp	308.6	1.20	0.0471
AFlow	157.9	3.20	0.0245
MaAS	4.8	1.20	0.0159
MAS^2	4.4	4.02	0.0011
OWL / Workforce	1198.2	53.73	0.1840
Agent-Co-op	401.6	3.22	0.0623
Ours	98.1	35.70	0.0127

Table 4: **Ablation of BDI components and intention commitment on ALFWorld.** Re-plan and Int. Switch directly measure deliberation stability. ΔSR is reported relative to the full model.

Variant	SR (%)	ΔSR (%p)	Tokens / task	Re-plan / task	Int. Switch / task
Component ablation					
w/o Belief update	11.94	−31.34	68,169	4.45	4.31
w/o Desire filtering	16.42	−26.87	61,614	6.21	6.07
w/o Intention commitment	31.34	−11.94	66,825	24.63	21.51
Ours (full)	**43.28**	–	75,901	2.50	2.43
Commitment strategy					
Blind commitment	38.81	−4.48	76,819	0.00	0.00
Single-minded (default)	**43.28**	–	75,901	2.50	2.43
Open-minded commitment	26.87	−16.42	83,802	18.82	18.64

candidate sub-goals are important for reliable orchestration. Removing intention commitment also lowers the success rate to 31.34%, while sharply increasing re-planning events from 2.50 to 24.63 and intention switches from 2.43 to 21.51 per task. This pattern is consistent with our motivation that commitment prevents the orchestrator from reconsidering its coordination strategy after every execution failure. Although this variant uses fewer tokens, its lower success rate indicates that token usage alone does not capture effective orchestration.

The commitment-strategy ablation further illustrates the trade-off between persistence and adaptability. Blind commitment, which never revises the intention, eliminates re-planning and intention switches but reduces the success rate to 38.81%. At the other extreme, open-minded commitment frequently revises the intention, resulting in 18.82 re-planning events and 18.64 intention switches per task, while achieving only 26.87% success. The default single-minded strategy achieves the highest success rate of 43.28% with substantially fewer revisions. These results suggest that neither rigid commitment nor frequent reconsideration is desirable; rather, preserving the current intention while allowing revision when plan-level adaptation is insufficient provides a more effective balance between commitment and adaptability.

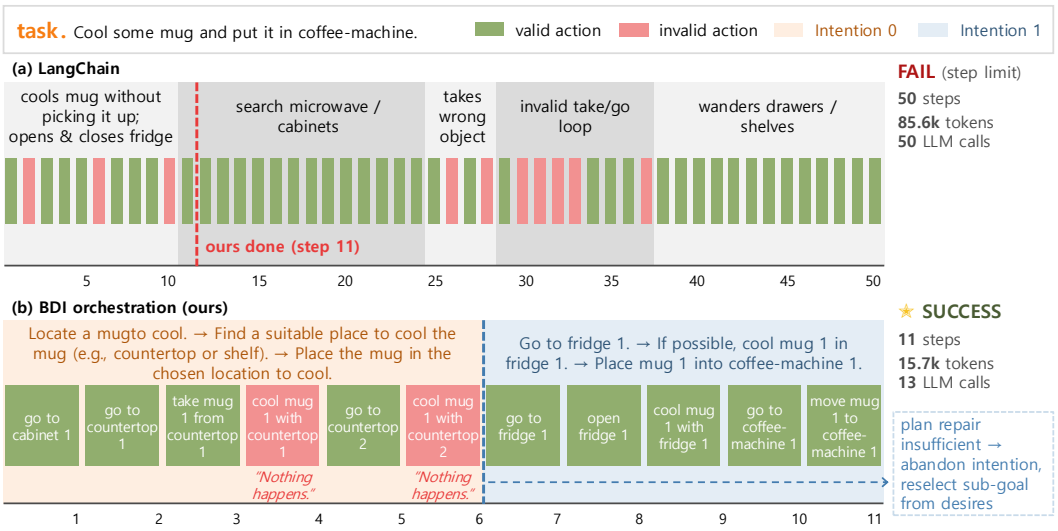

Figure 3: **Case study of BDI-based recovery from an ineffective plan.** The initial plan fails to make progress and reaches the step limit. The BDI orchestrator detects the failed execution, identifies a missing task prerequisite, and revises its intention accordingly. Guided by the revised intention, the agent replans its actions and successfully completes the task with a more goal-directed execution.

4.2.5 Case Study: Failure Recovery in ALFWorld

Figure 3 illustrates how the BDI orchestrator recovers from an ineffective initial plan. As shown in Figure 3(a), the initial plan repeatedly performs actions that do not lead to task completion and eventually reaches the maximum step limit. Rather than continuing the same ineffective behavior, the BDI orchestrator uses the execution outcome to reconsider its current intention and formulate a more appropriate sub-goal. As shown in Figure 3(b), this revised intention guides replanning toward the missing prerequisite: locating a suitable mug before attempting to make coffee. The resulting plan successfully completes the task in fewer steps, demonstrating how BDI-based reasoning enables the agent to recover from an ineffective plan by revising its intention and replanning accordingly.

5 Conclusion

In this paper, we propose a BDI-based orchestration framework for LLM multi-agent systems that adapts workflows during execution through Belief, Desire, and Intention. When a failure occurs, the framework keeps the current Intention as long as it remains achievable and revises the execution plan accordingly. The Intention is reconsidered only when plan-level adaptation is insufficient. Verified execution results are also retained as Beliefs so that previously obtained information can be reused during subsequent adaptation. This design separates plan adaptation from intention reconsideration and provides a way to determine what should be preserved and what should be changed during execution. The experimental and ablation results show that removing Belief updates, Desire filtering, or Intention commitment reduces performance, while removing intention commitment also increases re-planning and intention switching. These results suggest that maintaining the current Intention while revising the execution plan when necessary can serve as a practical approach to runtime orchestration in LLM multi-agent systems.

Limitations and Future Work Our framework has several limitations. Its improvement is limited on difficult GAIA Level 3 tasks, and step-wise execution and verification require a relatively large number of LLM calls. In addition, our experiments use a single backbone model. Future work will explore more efficient verification and adaptation mechanisms, stronger criteria for intention reconsideration, and evaluation across different backbone models and agentic environments.

AI USE STATEMENT

In this work, we used generative AI tools to aid and polish the writing, including improving grammar, clarity, and readability. We did not use generative AI tools to generate research ideas, conduct analyses, interpret results, or produce research findings. Other disclosure tasks are not applicable to this work. All AI-assisted text was reviewed and revised by the authors to ensure accuracy and consistency with the intended meaning. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REFERENCES

Michael E. Bratman. *Intention, Plans, and Practical Reason*. Harvard University Press, Cambridge, MA, 1987.

Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chi-Min Chan, Heyang Yu, Yaxi Lu, Yi-Hsin Hung, Chen Qian, Yujia Qin, Xin Cong, Ruobing Xie, Zhiyuan Liu, Maosong Sun, and Jie Zhou. Agentverse: Facilitating multi-agent collaboration and exploring emergent behaviors. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 20094–20136, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/578e65cdee35d00c708d4c64bce32971-Paper-Conference.pdf.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready ai agents with scalable long-term memory, 2025. URL https://arxiv.org/abs/2504.19413.

Runnan Fang, Yuan Liang, Xiaobin Wang, Jialong Wu, Shuofei Qiao, Pengjun Xie, Fei Huang, Huajun Chen, and Ningyu Zhang. Memp: Exploring agent procedural memory. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 17490–17502, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.866. URL https://aclanthology.org/2026.findings-acl.866/.

Adam Fourney, Gagan Bansal, Hussein Mozannar, Cheng Tan, Eduardo Salinas, Erkang Zhu, Friederike Niedtner, Grace Proebsting, Griffin Bassman, Jack Gerrits, Jacob Alber, Peter Chang, Ricky Loynd, Robert West, Victor Dibia, Ahmed Awadallah, Ece Kamar, Rafah Hosn, and Saleema Amershi. Magentic-one: A generalist multi-agent system for solving complex tasks, 2024. URL https://arxiv.org/abs/2411.04468.

Hongcheng Gao, Yue Liu, Yufei He, Longxu Dou, Chao Du, Zhijie Deng, Bryan Hooi, Min Lin, and Tianyu Pang. Reinforcing query-level meta-agents, 2026. URL https://openreview.net/forum?id=Tx9HKhGeQW.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. MetaGPT: Meta programming for a multi-agent collaborative framework. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=VtmBAGCN7o.

Mengkang Hu, Yuhang Zhou, Wendong Fan, Yuzhou Nie, Ziyu Ye, Bowei Xia, Tao Sun, Zhaoxuan Jin, Yingru Li, Zeyu Zhang, Yifeng Wang, Qianshuo Ye, Bernard Ghanem, Ping Luo, and Guohao Li. OWL: Optimized workforce learning for general multi-agent assistance in real-world task automation. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=MBJ46gd1CT.

Guohao Li, Hasan Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. Camel: Communicative agents for "mind" exploration of large language model society. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 51991–52008. Curran Associates, Inc., 2023. doi: 10.52202/075280-2264. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/a3621ee907def47c1b952ade25c67698-Paper-Conference.pdf.

Siru Ouyang, Jun Yan, I-Hung Hsu, Yanfei Chen, Ke Jiang, Zifeng Wang, Rujun Han, Long Le, Samira Daruki, Xiangru Tang, Vishy Tirumalashetty, George Lee, Mahsan Rofouei, Hangfei Lin, Jiawei Han, Chen-Yu Lee, and Tomas Pfister. Reasoningbank: Scaling agent self-evolving with reasoning memory. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 94327–94354, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/980ea04d23d1f6908964eba2a74afe45-Paper-Conference.pdf.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. Memgpt: Towards llms as operating systems, 2024. URL https://arxiv.org/abs/2310.08560.

Anand S. Rao and Michael P. Georgeff. Bdi agents: From theory to practice. In *Proceedings of the First International Conference on Multiagent Systems (ICMAS-95)*, pp. 312–319. AAAI Press, 1995.

Shuaike Shen, Wenduo Cheng, Shike Wang, Mingqian Ma, and Jian Ma. Agentco-op: Retrieval-based synthesis of interoperable multi-agent workflows, 2026. URL https://arxiv.org/abs/2605.20425.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: language agents with verbal reinforcement learning. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 8634–8652. Curran Associates, Inc., 2023. doi: 10.52202/075280-0377. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/1b44b878bb782e6954cd888628510e90-Paper-Conference.pdf.

Kun Wang, Guibin Zhang, ManKit Ye, Xinyu Deng, Dongxia Wang, Xiaobin Hu, Jinyang Guo, Yang Liu, and Yufei Guo. MAS^2: Self-generative, self-configuring, self-rectifying multi-agent systems. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=qumy27hMDY.

Yinjie Wang, Ling Yang, Guohao Li, Mengdi Wang, and Bryon Aragam. Scoreflow: Mastering llm agent workflows via score-based preference optimization, 2025. URL https://arxiv.org/abs/2502.04306.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W White, Doug Burger, and Chi Wang. Autogen: Enabling next-gen LLM applications via multi-agent conversations. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=BAakY1hNKS.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-mem: Agentic memory for llm agents. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 17577–17604. Curran Associates, Inc., 2025. doi: 10.52202/085713-0593. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/19909c36f51abc4856b4560aff3d36d6-Paper-Conference.pdf.

Rui Ye, Shuo Tang, Rui Ge, Yaxin Du, Zhenfei Yin, Siheng Chen, and Jing Shao. MAS-GPT: Training LLMs to build LLM-based multi-agent systems. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=3CiSpY3QdZ.

Guibin Zhang, Luyang Niu, Junfeng Fang, Kun Wang, LEI BAI, and Xiang Wang. Multi-agent architecture search via agentic supernet. In *Forty-second International Conference on Machine Learning*, 2025a. URL https://openreview.net/forum?id=imcyVlzpXh.

Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, XiongHui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, Bingnan Zheng, Bang Liu, Yuyu Luo, and Chenglin Wu. Aflow: Automating agentic workflow generation. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 34040–34077, 2025b. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/5492ecbce4439401798dcd2c90be94cd-Paper-Conference.pdf.

Zeyu Zhang, Quanyu Dai, Xiaohe Bo, Chen Ma, Rui Li, Xu Chen, Jieming Zhu, Zhenhua Dong, and Ji-Rong Wen. A survey on the memory mechanism of large language model-based agents. *ACM Transactions on Information Systems*, 43(6):1–47, 2025c. doi: 10.1145/3748302.

A BDI-AGENT IMPLEMENTATION DETAILS

A.1 BELIEF, DESIRE, INTENTION

- **Belief**: available agents and tools, evidence gathered during execution, and sub-task answers accepted by the verifier.
- **Desire**: the original query and its required answer format.
- **Intention**: a DAG of sub-tasks, each assigned to an agent.

A.2 CONTROL LOOP

The planner decomposes the query into an intention DAG. Sub-tasks are executed in dependency order, and each result is checked by a verifier. An accepted result is added to the belief and passed to the sub-tasks that depend on it. A rejected result triggers a repair. After all sub-tasks are finished, the final answer is produced from the accepted results.

A.3 VERIFICATION AND REPAIR

The verifier checks each result for completeness, grounding in evidence, and consistency with earlier accepted results. When a result is rejected, the agent escalates through a fixed sequence of repairs: RETRY $\rightarrow$ AUGMENT $\rightarrow$ REASSIGN $\rightarrow$ REPLAN. The sub-task is abandoned once this sequence is used up or the budget runs out. The choice of repair is made by rules. The LLM only writes the content of the repair.

A.4 NORM MEMORY

The agent records which repairs made a failed sub-task pass verification. On later tasks, it reuses repairs that have worked for similar failures. The memory is built only from the verifier's decisions and never uses ground-truth labels.

A.5 DATASET PROFILES

The control loop, verifier, and repair policy are shared across all datasets. Each dataset provides only its agent roster, its tools, and short planning notes.

A.6 INTERACTIVE ENVIRONMENTS

ALFWorld and TravelPlanner are multi-turn. For these, the belief is the interaction history, the desire is the task goal, and the intention is an ordered list of subgoals. The subgoal list is revised when progress stalls.

A.7 EXPERIMENTAL SETTINGS

All agents, and all baselines, use `gpt-4o-mini` at temperature 0. Every method is evaluated on the same tasks with the same scorer. Reported cost includes all LLM calls.

A.8 ABLATIONS

- **w/o belief update**: new evidence is not added to the belief.
- **w/o desire filtering**: no answer-format constraints are given.
- **w/o intention commitment**: the repair choice is reconsidered after every failure instead of escalating.