Rethinking Failure Attribution in Multi-Agent Systems: A Multi-Perspective Benchmark and Evaluation

Anonymous Authors[1]

Abstract

Failure attribution is essential for diagnosing and improving multi-agent systems (MAS), yet existing benchmarks and methods largely assume a single deterministic root cause for each failure. In practice, MAS failures often admit multiple plausible attributions due to complex inter-agent dependencies and ambiguous execution trajectories. We revisit MAS failure attribution from a multi-perspective standpoint and propose *multi-perspective failure attribution*, a practical paradigm that explicitly accounts for attribution ambiguity. To support this setting, we introduce MP-BENCH, the first benchmark designed for multi-perspective failure attribution in MAS, along with a new evaluation protocol tailored to this paradigm. Through extensive experiments, we find that prior conclusions suggesting LLMs struggle with failure attribution are largely driven by limitations in existing benchmark designs. Our results highlight the necessity of multi-perspective benchmarks and evaluation protocols for realistic and reliable MAS diagnosis.

1. Introduction

Multi-agent systems (MAS) enable multiple LLM-based agents to collaborate through complex interactions to solve tasks across diverse domains (Wu et al., 2024; Fourney et al., 2024; Hong et al., 2023; Ghafarollahi & Buehler, 2025; Zhang et al., 2024). Despite their increasing adoption, MAS continue to exhibit frequent execution failures, making it essential to understand where and why these failures occur (Cemri et al., 2025). To this end, recent work has focused on *failure attribution*, which aims to identify the failure-inducing steps from execution traces; however, existing approaches typically rely on manual, step-by-step inspection of agent behaviors, making them labor-intensive and time-consuming. To mitigate this burden, recent studies have explored automating failure attribution in MAS (Zhang et al.; 2025a;b; Kong et al., 2025; Ma et al., 2025; West et al., 2025). A pioneering effort is Who&When (Zhang et al.), the first benchmark for failure attribution in MAS, which labels each execution with a single, deterministic failure-inducing step for the task failure. Subsequent studies adopt this same formulation, inheriting the assumption that failure attribution can be uniquely determined (Zhang et al., 2025a; Kong et al., 2025).

However, we should rethink: *Is the failure-inducing step for a MAS failure deterministically decided?* Consider the example in Figure 1(a), where agents collaborate to address the user query. The Who&When dataset and subsequent works implicitly assume that the oracle of failure-inducing steps is deterministic, thereby only one failure step are decided during annotation process (Figure 1(b)).

In practice, the failure-inducing step of a MAS failure is often **not deterministic and admits multiple plausible attributions**. As illustrated in Figure 1(c), different analysts may reasonably attribute the failure to different steps. This non-deterministic and multi-perspective nature arises from the intrinsic interdependencies of MAS. Multiple specialized agents can be interdependently composed into diverse, valid execution trajectories that aim toward the same goal. Consequently, failure attribution depends heavily on an observer's internal reasoning of what constitutes a "correct" trajectory. In Figure 1(c), the perspective difference emerges because one analyst expects a web-first strategy, while another considers a code-first approach acceptable. As further demonstrated in Section 4.3, even expert annotators exhibits perspective differences on the failure-inducing step in up to about 60% of cases.

However, existing works largely overlook the inherently multi-perspective nature of MAS failure attribution, limiting their practical applicability. Since current benchmarks are built upon the assumption that the the oracle of a failure-inducing step is deterministic, they often misjudge reasonable predictions as incorrect, undermining evaluation reliability. Furthermore, models built on the assumption inherit biases that fail to capture diverse causal patterns, hindering

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

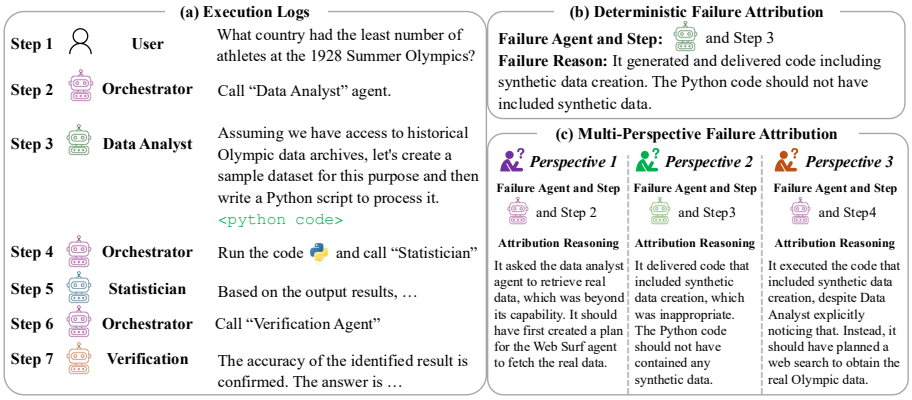

Figure 1. Motivating examples illustrating the multi-perspective nature of MAS failure attribution. (a) presents a simplified execution log of a MAS. (b) presents the example of deterministic failure attribution existing approaches assume (c) presents the example of multi-perspective failure attribution.

effective diagnosis of MAS.

To address these limitations, we propose a more practical paradigm: ***multi-perspective failure attribution*** for MAS. Under this paradigm, a failure attribution system identifies failure-inducing steps from multiple perspectives, each accompanied by explicit internal reasoning that explains the system's judgment from that viewpoint. In contrast to prior work that frames failure attribution as a rigid, deterministic labeling task, our paradigm yields the multi-perspective attributions with concrete reasoning and actionable signals that more effectively support systematic MAS failure diagnosis.

This new paradigm necessitates a corresponding benchmark dataset and evaluation protocol. To this end, we introduce a new benchmark, **m**ulti-**p**erspective failure attribution **bench** (MP-BENCH), consisting of execution logs collected from diverse tasks and MAS configurations. Each log is annotated by multiple expert annotators who are rigorously screened via technical interviews and preliminary quality assessments on held-out test cases. The annotations include the failure-inducing step, and explicit internal reasoning that justifies the attribution from each annotator's perspective. This expert-driven annotation process provides substantially stronger reliability than prior benchmarks, which largely rely on automated pipelines. We further design an evaluation protocol to assess whether current LLMs can perform multi-perspective failure attribution on MP-BENCH. The evaluation focuses on two core capabilities: (1) the ability to identify failure-inducing steps from multiple perspectives, and (2) the ability to provide reliable and well-grounded reasoning to justify these attributions.

Through extensive experiments with state-of-the-art LLMs, we show that when sampled multiple times, LLMs effectively identify diverse failure attributions that correspond to different perspectives. Under our evaluation protocol, these attributions and the associated reasoning align closely with human expert's internal reasoning. Notably, these findings contrast sharply with prior results reported in deterministic benchmarks such as Who&When, where LLM-based step-level failure attribution was shown to perform near random. Together, our results demonstrate that the limitations of LLMs in failure attribution observed in existing works stem not from insufficient model capability, but from biased evaluation assumptions that overlook the inherently multi-perspective nature of MAS failures. Moreover, we provide practical guidance for real-world MAS failure attribution: leveraging diverse sampling and complementary perspectives across models can more effectively capture diverse causal failure patterns and support systematic system diagnosis. The key contributions of this work are as follows:

- We identify a fundamental assumption in prior work that limits the practicality of MAS failure attribution and propose a more realistic task formulation, *multi-perspective failure attribution*.
- We introduce a new benchmark and evaluation protocol specifically designed for multi-perspective failure attribution in MAS.
- We show that the prevailing belief that current LLMs perform poorly at failure attribution arises from flawed deterministic assumptions in existing benchmarks, rather than from inherent limitations of the models themselves.
- We provide practical guidance for real-world MAS failure attribution, demonstrating how leveraging diverse sampling strategies and complementary model perspectives can lead to more effective system diagnosis.

2. Related Works

2.1. LLM-based Multi-Agent Systems

Multi-agent systems built on LLMs have recently emerged as an effective paradigm for solving complex tasks through

collaboration among multiple specialized agents (Hong et al., 2023; Wu et al., 2024; Zhang et al., 2024; Song et al., 2024; Ye et al., 2025). While such systems benefit from modularity and role specialization, their intricate inter-agent dependencies often introduce reliability issues, where errors propagate across agents and failures arise from complex interactions (Cemri et al., 2025; Zhang et al.). These challenges become more severe in long-horizon, multi-step reasoning settings, highlighting the need for systematic analysis of where and why MAS fail. Accordingly, this work focuses on failure attribution in MAS.

2.2. Benchmarks for Failure Attribution of MAS

Automatic failure attribution in MAS identifies failure-inducing steps from execution traces to support MAS diagnosis and refinement; accurately evaluating such systems is therefore essential. While several benchmarks have recently attracted attention, they remain limited in practicality for benchmarking real-world failure attribution systems. For example, MAST (Cemri et al., 2025) is restricted to execution-level failure mode analysis and thus cannot evaluate step-level failure attribution. TracerTraj (Zhang et al., 2025a) and Aegis (Kong et al., 2025) pursue finer-grained step- or agent-level attribution; however, their benchmarks are constructed through fully automated pipelines, raising concerns about annotation reliability given the complexity of MAS logs and causal failure patterns. Who&When (Zhang et al.) addresses these limitations by employing expert annotators to construct the benchmark dataset. Nevertheless, these benchmarks adopt a single-perspective formulation that assumes a deterministic oracle failure-inducing step, an assumption that conflicts with real-world MAS failures where multiple plausible attributions often coexist depending on an observer's perspective. Moreover, they frame failure attribution as a rigid labeling task and do not support evaluating the reasoning behind an attribution, which is essential for systematic failure diagnosis.

Table 1. Comparison between our benchmark and others.

	Attribution Granularity	Annotation Method	Attribution Reasoning Eval	Perspective	Size
Cemri et al. (2025)	Execution	Hybrid	✗	Single	1,600
Zhang et al.	Step	Expert	✗	Single	184
Zhang et al. (2025a)	Step	Automated	✗	Single	2,500
Kong et al. (2025)	Agent	Automated	✗	Single	9,500
MP-BENCH	Step	Expert	✓	Multi	289

To address these limitations, we introduce the first multi-perspective failure attribution benchmark for MAS, constructed through a rigorous expert annotator recruitment and labeling process to ensure consistently high-quality annotations. We further propose a new evaluation protocol tailored to the multi-perspective setting, enabling systematic comparison of failure attribution systems under attribution ambiguity. In addition, our benchmark provides annotator rationales for each attribution, along with an evaluation framework that explicitly assesses both attribution quality and underlying reasoning. It enables structured and fine-grained analysis of failure patterns across diverse agents, tasks, and perspectives. A detailed comparison between existing benchmarks and our contributions is summarized in Table 1.

2.3. Annotator Disagreement

Recent literature has increasingly adopted a perspectivist approach that treats annotator disagreement as a meaningful signal rather than noise to be resolved (Fleisig et al., 2024; Wan et al., 2025). Prior work has explored various modeling strategies to preserve and leverage diverse viewpoints (Dehghan et al., 2025; Xu et al., 2025; Homayounirad et al., 2025). However, these studies have largely focused on socially subjective tasks—such as hate speech or radical content detection (Dehghan et al., 2025; Riabi et al., 2025; Orlikowski et al., 2025; Fleisig et al., 2023)—where judgments are often grounded in personal, cultural, or normative values rather than technical reasoning.

In contrast, we present the first study to investigate annotator disagreement in the context of failure attribution for MAS executions, a setting that requires complex technical reasoning and system diagnosis. Moreover, we introduce a novel benchmark that explicitly captures the multi-perspective nature of MAS failures. By doing so, we extend the scope of perspectivism beyond social subjectivity to address ambiguity arising from the interpretability and diagnostic complexity of real-world MAS.

3. Task Formulation

3.1. Preliminaries

Multi-Agent Systems. We adopt a widely used turn-based LLM multi-agent protocol (Hong et al., 2023; Li et al., 2023; Wu et al., 2024; Zhang et al.). Specifically, we consider an LLM-powered MAS $\mathcal{M}$ consisting of a set of N agents, denoted by $\mathcal{N} = 1, 2, \ldots, N$, that operate in discrete time steps. At each time step, exactly one agent takes an action according to a turn-based interaction rule. The system is formally defined as $\mathcal{M} = \langle \mathcal{N}, S, A, P, \phi \rangle$, where S denotes the set of environment states, A is the global action space, and each agent $i \in \mathcal{N}$ is associated with a subset of allowable actions $A_i \subseteq A$. The function $\phi(t)$ specifies the active agent at time step t. The state transition dynamics are governed by $P(s_{t+1} \mid s_t, a_t, \phi(t))$, which defines the probability of transitioning to state s_{t+1} given the observable state s_t and the action a_t taken by the active agent at time t. An execution of the system over a task is represented as a finite trajectory $\tau = (s_1, a_1, \phi(1)), \ldots, (s_T, a_T, \phi(T))$, where T denotes the total number of steps.

Deterministic Failure Attribution. Existing failure attribu-

tion works assume that, for each failed execution τ, there exists a deterministic set of failure-inducing steps that can be uniquely identified. Under this assumption, failure attribution is formulated as a mapping $f_{\text{det}}(\tau) \rightarrow \mathcal{F}^*$, where $\mathcal{F}^* \subseteq \{1, \ldots, T\}$ denotes a fixed set of time steps whose associated agent actions $(a_t, \phi(t))$ are deemed responsible for the failure. This formulation treats failure attribution as an objective labeling problem with a single correct attribution set. In practice, most benchmarks further simplify this formulation by assuming $|\mathcal{F}^*| = 1$, selecting a single step as failure-inducing (Zhang et al.; 2025a).

3.2. Multi-Perspective Failure Attribution

Definition. In contrast, we argue that MAS failure attribution is inherently non-deterministic. Due to the existence of multiple valid execution trajectories and diverse expectations of a correct strategy, a failed execution may admit multiple plausible failure attributions, each grounded in a distinct analytical perspective.

We define *multi-perspective failure attribution* as the task of identifying a set of plausible failure-inducing steps for a given failed execution τ, together with their associated rationales. Formally, the output is a set $\mathcal{F}(\tau) = \{(t_k, r_k)\}_{k=1}^{K}$, where each $t_k \in \{1, \ldots, T\}$ denotes a failure-inducing time step, and r_k is an explicit rationale explaining why the agent action $(a_{t_k}, \phi(t_k))$ is considered a failure-inducing step under a particular perspective k. The cardinality K is not fixed and may vary across executions. Importantly, multiple attributions in $\mathcal{F}(\tau)$ can be simultaneously valid, even when they identify different steps—and thus different agents via $\phi(t_k)$—as responsible for the failure.

Task Objective. Given a failed execution trace τ, the objective of multi-perspective failure attribution is to predict a set of plausible failure attributions $\widehat{\mathcal{F}}(\tau)$ that aligns with human expert judgments. Unlike deterministic settings, success is not defined by matching a single oracle label, but by the model's ability to recover multiple reasonable attributions and their associated rationales.

This formulation explicitly acknowledges the multi-perspective nature of MAS failure attribution and reframes the task from identifying a single failure-inducing step to characterizing a space of plausible causal explanations.

4. MP-BENCH: Multi-Perspective Failure Attribution Benchmark

4.1. Benchmark Dataset Construction

This section describes the construction of MP-BENCH, a benchmark dataset designed for *multi-perspective failure attribution* in MAS. Our dataset is built to capture diverse failure patterns across systems, tasks, and annotator perspectives, while providing structured rationales for the step-level agent actions. The overall procedure is summarized in Figure 2.

4.1.1. MULTI-AGENT SYSTEMS AND EXECUTION LOG COLLECTION

To ensure diversity in system design and execution behavior, following (Zhang et al.), we include both hand-crafted and fully-automated MAS configurations. Specifically, we collect 289 logs from a total of 121 distinct MAS configurations and various tasks. For hand-crafted MAS, we use MAgenticOne (Fourney et al., 2024). For fully-automated MAS, we use CaptainAgent (Song et al., 2024). For tasks, we use GAIA (Mialon et al., 2023) and AssistantBench (Yoran et al., 2024), which are widely utilized to evaluate the agentic capabilities of LLM-based MAS.

As the first benchmark to address multi-perspective failure attribution, we prioritize the quality of annotation over dataset scale. We argue that for evaluating complex diagnostic capabilities such as failure attribution—where understanding nuanced causal patterns and diverse interpretations is critical—a carefully curated set of expert annotations provides more reliable evaluation signals than automated labels that may lack grounding or consistency.

While our dataset of 289 execution logs is smaller than recent automated benchmarks (e.g., TracerTraj with 2,500 instances and Aegis with 9,500), these works rely on fully automated data generation pipelines that cannot ensure annotation reliability for the complex reasoning patterns inherent in MAS failures. In contrast, we employ a rigorous expert annotation process with multiple annotators per instance, capturing diverse perspectives and explicit reasoning for each attribution. Compared to Who&When, the closest prior work that also uses manual curation, our benchmark provides 57% more instances (289 vs. 184) while additionally introducing the multi-perspective paradigm. This design enables broader coverage of failure patterns and more realistic evaluation of attribution systems in practice.

4.1.2. EXPERT ANNOTATOR RECRUITMENT

We recruit three expert annotators on Upwork[1] platform. To ensure high-quality and technically grounded annotations, we prioritize candidates with professional experience in LLMs, AI systems, or agent-based frameworks. We adopt a rigorous two-stage screening process. In the first stage, we conduct technical interviews to assess candidates' conceptual understanding of the task. Applicants are asked to explain, in their own words, what MAS failure attribution entails, and to describe how they would approach annotating a turn-based MAS execution log. In the second stage, we

[1] https://www.upwork.com/

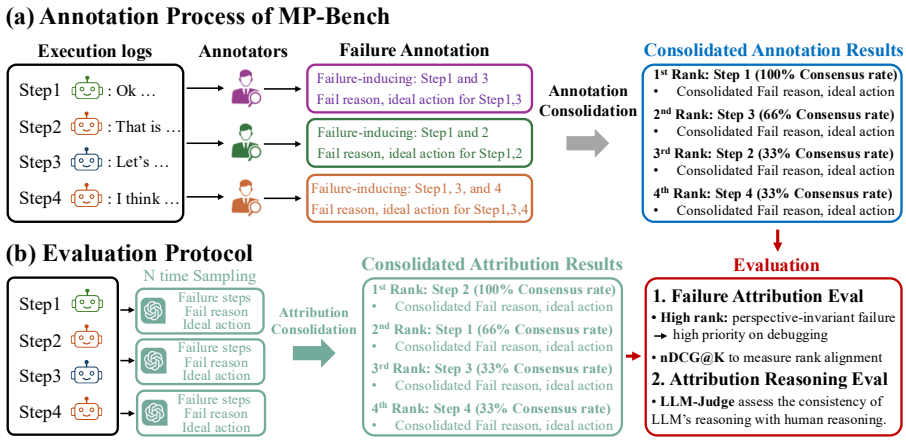

Figure 2. Overall framework of the (a) annotation process and (b) evaluation protocol of MP-BENCH.

assess annotation quality using test cases. Specifically, we provide several MAS execution logs and ask candidates to annotate them according to a detailed predefined task description (see Figure 6 in Appendix). The authors manually review the submitted annotations, focusing on correctness, consistency, and the clarity of reasoning. Only candidates who pass both stages are selected, resulting in three verified annotators.

4.1.3. ANNOTATION TASK

For each execution log, annotators are provided with the complete MAS execution trace and task description. Each trace consists of a sequence of agent actions following a turn-based protocol. Annotators independently annotate every step, assigning the following for each:

- **Failure-inducing Step:** a binary label indicating whether the step is failure-inducing.
- **Failure Reason:** a free-text explanation justifying the failure annotation.
- **Ideal Action:** a free-text description of the action that the agent should ideally have taken.

Critically, we provide no potential bias or guiding perspectives to annotators, allowing them to develop independent interpretations based on their expertise. This design genuinely captures the multi-perspective nature of MAS failure attribution, as annotators may reasonably disagree on which steps are failure-inducing based on different assumptions about correct execution strategies. The *failure reason* and *ideal action* fields explicitly capture each annotator's internal reasoning, grounding why a particular agent action is deemed problematic from their perspective. This enables MP-BENCH to evaluate failure attribution systems on two dimensions: (1) the ability to identify failure-inducing steps across multiple perspectives, and (2) the ability to generate well-grounded reasoning that aligns with expert judgment. This evaluation framework moves beyond existing benchmarks that treat failure attribution as deterministic classification, instead assessing both attribution accuracy and reasoning quality.

4.1.4. ANNOTATION CONSOLIDATION

After collecting annotations from all annotators, we consolidate them into a ranking-based format. For each execution log, we compute a *consensus rate* for every step, defined as the fraction of annotators who labeled it as failure-inducing. Steps are then ranked by consensus rate. Steps with 100% consensus represent perspective-invariant failure that is consistently recognized across all annotators. Such steps represent the most salient and unambiguous failure points, and therefore should be assigned the highest priority during MAS refinement. In contrast, steps labeled as failures by only a subset of annotators reflect perspective-dependent failure modes. These steps may be considered failures under certain interpretations of a correct execution trajectory, but not under others, and thus typically warrant lower refinement priority. Importantly, only steps that are labeled as failures by at least one annotator are included in the ranking; steps that are never identified as failures are excluded. By ranking failure-inducing steps according to their consensus rates, our formulation transforms multi-perspective annotations into an actionable prioritization signal for developers, aligning failure attribution with practical MAS diagnosis workflows.

While binary failure annotations naturally aggregate into consensus-based rankings, failure reasons and ideal actions are free-text and require different treatment. We consolidate these textual explanations using LLM-assisted summarization. Specifically, we prompt an LLM to synthesize a unified failure reason and ideal action for each failure-inducing step, conditioned on all annotator-provided explanations. The LLM is instructed to comprehensively cover the di-

verse rationales without introducing new information. We primarily use GPT-5.1 (OpenAI, 2025c) for consolidation. To verify robustness to this choice, we additionally evaluate consolidations generated by GPT-4.1 (OpenAI, 2025b) and Claude-Sonnet-4.5 (OpenAI, 2025a). We observe that model rankings and performance patterns remain consistent across all three consolidation methods, confirming that our results are not sensitive to the choice of consolidation model. Detailed results are presented in Appendix E.1.

4.2. Annotation Quality and Effort

Details of the annotators and annotation effort are provided in Appendix C.

4.3. Data Analysis

We analyze the annotation statistics of MP-BENCH, focusing on the degree of consensus among annotators regarding failure step annotations. As shown in Figure 3(a), only 16.2% of steps are annotated as failure-inducing by all three annotators. In contrast, 27.8% of steps are labeled as failures by exactly two annotators, while a substantial 56.1% are labeled as failures by only a single annotator. This distribution indicates that failure attribution is highly dependent on annotator perspective, underscoring the limitations of deterministic assumptions adopted by existing benchmarks. Similarly, Figure 3(b) reports inter-annotator disagreement rates, defined as cases where one annotator marks a step as failure while another does not. The results show that the disagreement can reach up to 60% across annotators, further confirming the multi-perspective nature of MAS failure attribution.

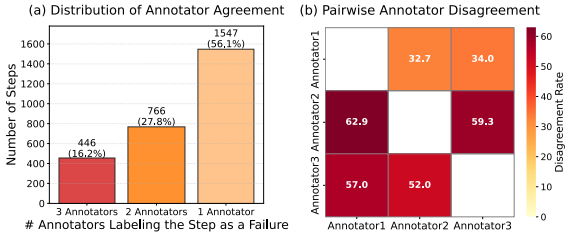

Figure 3. Analysis of MP-BENCH highlighting the multi-perspective nature of MAS failure attribution. (a) Distribution of steps grouped by annotator consensus on failure annotations. (b) Inter-annotator disagreement rates for failure annotations.

5. Evaluation Protocol

This section describes the evaluation protocol for benchmarking MAS failure attribution systems on MP-BENCH. The protocol is explicitly designed to address the following questions:

- Do LLM-based failure attribution systems capture failures from multiple perspectives? (Section 5.1)
- Do LLMs provide reliable reasoning for failure attribution? (Section 5.2)

An overview of the evaluation protocol is shown in Figure 2(b).

5.1. Failure Attribution Evaluation

Existing failure attribution benchmarks formulate evaluation as a classification task, assessing whether a system correctly identifies a single failure-inducing step. Such deterministic evaluation is insufficient in our setting, as a single execution may admit multiple plausible failure attributions depending on perspective.

To address the first question, in MP-BENCH, we adopt a ranking-based evaluation that measures the consistency between the rankings produced by LLM-based failure attribution systems and the ground-truth rankings derived from human annotations. Specifically, we run each LLM N times with a stochastic sampling, generating multiple independent failure attribution outputs. Each output consists of predicted failure-inducing steps along with their associated failure reasons, and ideal actions. Following the same consolidation procedure used for human annotations, we aggregate these outputs into a single predicted ranking based on consensus rates.

To compare rankings produced by annotators and LLMs, we employ normalized Discounted Cumulative Gain (nDCG@K). This metric evaluates the agreement between the predicted ranking and the ground-truth ranking, with greater emphasis on correctly prioritizing highly salient failure steps. As a result, nDCG@K aligns evaluation with practical MAS diagnosis, where addressing the most critical failures first is essential.

5.2. Attribution Reasoning Evaluation

In the multi-perspective failure attribution task, beyond identifying failure-inducing steps, LLMs generate explicit attribution reasoning including failure reasons and ideal actions. Evaluating this reasoning quality is essential for assessing whether LLMs can serve as reliable failure attribution systems.

Following the same procedure as annotation consolidation (Section 4.1.4), we consolidate the attribution reasoning generated from N LLM runs. We then adopt an LLM-as-a-Judge framework using GPT-5.1 as the judge model. For each failure-inducing step, the judge assesses consolidated predictions against consolidated ground truths based on: (1) consistency with human judgments, (2) grounding in execution context, (3) explanatory adequacy, and (4) reasonableness of proposed actions. The judge assigns scores (1-10) for failure reason, ideal action, and overall reasoning quality. We calculate the average score of overall reason-

Table 2. Failure attribution and attribution reasoning performance comparison.

	Hand-Crafted MAS					Automatic MAS				
	Failure Attribution Eval				Attr.	Failure Attribution Eval				Attr
	nDCG@5		nDCG@full		Reason.	nDCG@5		nDCG@full		Reason.
Model	Exp	Lin.	Exp	Lin.	Score	Exp	Lin.	Exp	Lin.	Score
Random	0.128	0.237	0.148	0.223	-	0.315	0.415	0.321	0.42	-
GPT-4.1	0.431	0.539	0.472	0.496	7.63	0.676	0.695	0.676	0.685	7.78
GPT-5.1	0.375	0.516	0.453	0.550	7.62	0.784	**0.813**	0.786	**0.805**	**8.3**
O3-mini	0.437	0.530	0.449	0.444	7.48	0.394	0.409	0.394	0.398	7.37
Sonnet-4.5	**0.440**	**0.581**	**0.516**	**0.57**	**7.95**	**0.790**	0.800	**0.792**	0.794	8.23
Qwen3-8B	0.294	0.395	0.338	0.389	6.11	0.668	0.674	0.681	0.678	6.54
oss-120B	0.425	0.546	0.485	0.541	7.51	0.703	0.717	0.713	0.717	7.92

ing quality across all steps and executions. Detailed judge prompts are provided in Figure 8 in Appendix.

To verify robustness to judge model choice, we additionally evaluate using GPT-4.1 and Claude-Sonnet-4.5 as alternative judges. We find that model rankings remain highly consistent across all three judges, confirming that our evaluation results are not sensitive to this choice. Detailed results are presented in Appendix E.1.

6. Experiment

6.1. Experimental Details

For failure attribution systems, we adopt the All-at-Once method (Zhang et al.) with a diverse set of LLMs. Detailed implementation and evaluation settings are provided in Appendix D.1 and Appendix D.2, respectively.

6.2. Experimental Results

Main Results. From the evaluation on MP-BENCH, we derive the following key observations.

First, stochastic sampling of LLMs naturally reflects diverse perspectives in failure attribution. When sampling multiple failure attributions from LLMs, we observe substantial disagreement among the predicted failure-inducing steps (e.g., GPT-5.1 in Figure 4(a)), a pattern that closely mirrors the disagreement observed among human expert annotators. This alignment provides further evidence that failure attribution in MAS is inherently non-deterministic and admits multiple plausible perspectives, rather than a single deterministic root cause.

Second, under a multi-perspective evaluation setting, LLMs perform failure attribution effectively when sampled multiple times. As shown in Table 2, when LLMs are sampled multiple times ($N = 3$), they successfully identify multiple reasonable failure-inducing steps across different perspectives. Among the evaluated models, Claude-Sonnet-4.5 consistently achieves the strongest performance, whereas Qwen3-8B exhibits limited failure attribution and reasoning capabilities, likely due to its relatively small

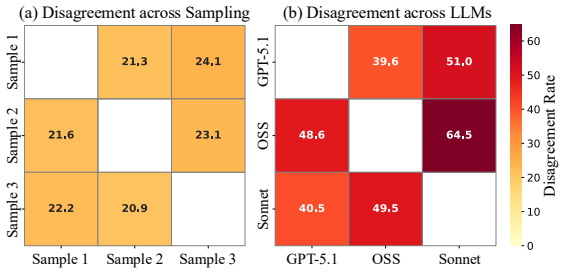

Figure 4. Disagreement analysis of the LLM-based failure attribution system. (a) Pairwise disagreement rates across different samplings of GPT-5.1. (b) Pairwise disagreement rates across different LLMs.

model size. Notably, we observe that stronger reasoning-oriented or more recent models do not necessarily guarantee superior performance in this setting. That said, the resulting nDCG scores from all models are substantially higher than those of a random baseline. These results stand in contrast to prior findings from deterministic benchmarks such as Who&When (Zhang et al.), which suggested that LLMs struggle with step-level failure attribution by exhibiting near-random performance.

Third, LLM-generated failure attributions are accompanied by reliable and well-grounded reasoning. In Table 2, we observe high LLM-as-a-Judge scores when comparing the attribution reasoning produced by LLMs with that of human annotators. This indicates that LLM-generated multi-perspective attributions are not only plausible in terms of identified failure steps, but are also supported by reasoning that aligns well with human expert judgments. Consequently, these attributions can serve as meaningful diagnostic signals for MAS developers.

Overall, our results challenge the prevailing assumption that MAS failure attribution is deterministic and that in-context LLMs are inherently ineffective at this task. Instead, we show that MAS failure attribution exhibits an inherently multi-perspective nature, and that LLMs possess sufficient capability to capture this structure when evaluated under an appropriate benchmark and evaluation protocol. The limitations reported in prior work therefore stem not from deficiencies in model capability, but from biased evaluation assumptions that overlook the multi-perspective nature of MAS failures.

6.2.1. SENSITIVITY ANALYSES ON NUMBER OF LLM SAMPLING

We further examine the impact of the number of LLM runs (N) on multi-perspective failure attribution performance. We vary N from 3 to 10 and report nDCG@5 with exponential gain for different LLMs. As shown in Table 3, increasing N leads to consistent performance improvements

Table 3. Performance comparison (nDCG@5 with exponential gain) across different number of LLM runs N. Bold values indicate the best performance within each model.

	Hand-Crafted (N)			Automatic (N)		
	3	5	10	3	5	10
GPT-4.1	0.436	0.438	**0.462**	0.687	0.695	**0.720**
GPT-5.1	0.381	0.399	**0.420**	**0.782**	0.780	0.777
O3-mini	0.432	0.473	**0.486**	0.413	0.425	**0.460**
Sonnet-4.5	0.447	0.467	**0.475**	0.766	0.776	**0.786**
Qwen3-8B	0.295	0.324	**0.344**	0.662	0.690	**0.723**
oss-120B	0.422	0.457	**0.478**	0.712	0.719	**0.727**

across models. Notably, performance continues to improve as N increases, indicating that aggregating a larger number of samples yields more stable and reliable failure attribution rankings. This trend suggests that multiple runs help mitigate variability arising from limited sampling and enable more robust aggregation of perspective-dependent failure signals.

6.2.2. Analyses on Multi-LLM Combination

Table 4. Performance comparison between single-LLM and multi-LLM failure attribution systems. For brevity, we denote GPT-4.1, GPT-5.1, GPT-OSS-120B, and Claude-Sonnet-4.5 as G-4.1, G-5.1, OSS, and Sonnet, respectively.

Model Combination				nDCG@5 (Exp)	
G-4.1	G-5.1	OSS	Sonnet	Hand.	Auto.
✓	✗	✗	✗	0.431	0.676
✗	✓	✗	✗	0.375	0.784
✗	✗	✓	✗	0.425	0.703
✗	✗	✗	✓	0.440	0.789
✓	✓	✓	✗	0.480	0.800
✓	✓	✗	✓	0.491	0.814
✓	✗	✓	✓	**0.496**	0.794
✗	✓	✓	✓	0.491	**0.823**

We study whether collaborating multiple LLMs improves failure attribution performance, motivated by the hypothesis that different models encode distinct reasoning patterns and thus offer complementary attribution perspectives. To evaluate this, we compare single-LLM systems with multi-LLM systems constructed from GPT-4.1, GPT-5.1, GPT-OSS-120B, and Claude-Sonnet-4.5. We focus on combinations that achieve strong performance in Table 2. In the single-LLM setting, each model is sampled three times ($N = 3$), whereas in the multi-LLM setting, each model is sampled once. All results are evaluated using nDCG@5 with exponential gain.

The results in Table 4 show that multi-LLM systems consistently outperform their single-LLM counterparts. Importantly, the largest performance gains are obtained when combining models from different families. For instance, mixtures of Claude, GPT-based, and OSS models achieve the highest attribution performance, suggesting that cross-family diversity is particularly effective for failure attribution.

This observation is further supported by the disagreement analysis in Figure 4. We find that pairwise disagreement across multiple samplings of the same LLM is substantially lower than the disagreement observed across different LLMs. This indicates that repeated sampling primarily captures stochastic variation, whereas combining distinct models introduces qualitatively different attribution perspectives. Taken together, these findings suggest that the performance gains of multi-LLM systems stem from complementary internal reasoning patterns across models, rather than from increased sampling alone.

6.3. Practical Guidance for Practitioners

Based on our experimental findings, we provide practical guidance for designing effective and reliable automatic failure attribution systems for MAS.

First, MAS failure attribution is inherently multi-perspective. Rather than relying on a single deterministic attribution result, practitioners should emphasize aggregating attributions from multiple perspectives, which enables capturing diverse causal patterns and supports more effective MAS diagnosis and refinement.

Second, combining LLMs from different model families is strongly recommended. Models from different families tend to provide complementary perspectives, and aggregating their attributions leads to more informative and robust failure diagnosis.

Third, when higher reliability is required, increasing the number of samples is beneficial. Aggregating a larger set of attribution results enables more stable and accurate prioritization of failure-inducing steps.

7. Conclusion

We revisit failure attribution in MAS and show that deterministic formulations fail to capture the inherently multi-perspective nature of real-world MAS failures. To address this limitation, we propose multi-perspective failure attribution and introduced MP-Bench, the first benchmark and evaluation framework designed for this setting. MP-Bench combines expert annotations and multi-perspective evaluation to enable faithful assessment of failure attribution quality and reasoning. Our experiments reveal that prior claims about LLMs' limitations in failure attribution largely stem from restrictive benchmark assumptions, and that modern LLMs can produce diverse, human-aligned attributions when evaluated appropriately. Overall, our work underscores the need to rethink failure attribution benchmarks and evaluation practices, and provides a practical foundation for reliable MAS diagnosis.

References

Agarwal, S., Ahmad, L., Ai, J., Altman, S., Applebaum, A., Arbus, E., Arora, R. K., Bai, Y., Baker, B., Bao, H., et al. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*, 2025.

Anthropic. Anthropic api. `https://www.anthropic.com/api/`.

Cemri, M., Pan, M. Z., Yang, S., Agrawal, L. A., Chopra, B., Tiwari, R., Keutzer, K., Parameswaran, A., Klein, D., Ramchandran, K., et al. Why do multi-agent llm systems fail? *arXiv preprint arXiv:2503.13657*, 2025.

Dehghan, S., Sen, M. U., and Yanikoglu, B. Dealing with annotator disagreement in hate speech classification. *arXiv preprint arXiv:2502.08266*, 2025.

Fleisig, E., Abebe, R., and Klein, D. When the majority is wrong: Modeling annotator disagreement for subjective tasks. *arXiv preprint arXiv:2305.06626*, 2023.

Fleisig, E., Blodgett, S. L., Klein, D., and Talat, Z. The perspectivist paradigm shift: Assumptions and challenges of capturing human labels. *arXiv preprint arXiv:2405.05860*, 2024.

Fourney, A., Bansal, G., Mozannar, H., Tan, C., Salinas, E., Niedtner, F., Proebsting, G., Bassman, G., Gerrits, J., Alber, J., et al. Magentic-one: A generalist multi-agent system for solving complex tasks. *arXiv preprint arXiv:2411.04468*, 2024.

Ghafarollahi, A. and Buehler, M. J. Sciagents: automating scientific discovery through bioinspired multi-agent intelligent graph reasoning. *Advanced Materials*, 37(22): 2413523, 2025.

Homayounirad, A., Liscio, E., Wang, T., Jonker, C. M., and Siebert, L. C. Will annotators disagree? identifying subjectivity in value-laden arguments. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 15237–15252. Association for Computational Linguistics, 2025.

Hong, S., Zhuge, M., Chen, J., Zheng, X., Cheng, Y., Wang, J., Zhang, C., Wang, Z., Yau, S. K. S., Lin, Z., et al. Metagpt: Meta programming for a multi-agent collaborative framework. In *The Twelfth International Conference on Learning Representations*, 2023.

Kong, F., Zhang, R., Yin, H., Zhang, G., Zhang, X., Chen, Z., Zhang, Z., Zhang, X., Zhu, S.-C., and Feng, X. Aegis: Automated error generation and attribution for multi-agent systems. *arXiv preprint arXiv:2509.14295*, 2025.

Li, G., Hammoud, H., Itani, H., Khizbullin, D., and Ghanem, B. Camel: Communicative agents for" mind" exploration of large language model society. *Advances in Neural Information Processing Systems*, 36:51991–52008, 2023.

Ma, G., Zhu, J., Guo, H., Shi, W., Shen, J., Liu, J., and Liang, Y. Automatic failure attribution and critical step prediction method for multi-agent systems based on causal inference. *arXiv preprint arXiv:2509.08682*, 2025.

Mialon, G., Fourrier, C., Wolf, T., LeCun, Y., and Scialom, T. Gaia: a benchmark for general ai assistants. In *The Twelfth International Conference on Learning Representations*, 2023.

OpenAI. Openai api. `https://openai.com/index/openai-api/`.

OpenAI. Introducing claude sonnet 4.5, 2025a. URL `https://www.anthropic.com/news/claude-sonnet-4-5`.

OpenAI. Introducing gpt-4.1 in the api, 2025b. URL `https://openai.com/index/gpt-4-1/`.

OpenAI. Introducing gpt-5.1 in the api, 2025c. URL `https://openai.com/index/gpt-5-1/`.

OpenAI. Openai o3-mini system card, 2025d. URL `https://openai.com/index/o3-mini-system-card/`.

Orlikowski, M., Pei, J., Röttger, P., Cimiano, P., Jurgens, D., and Hovy, D. Beyond demographics: Fine-tuning large language models to predict individuals' subjective text perceptions. *arXiv preprint arXiv:2502.20897*, 2025.

Riabi, A., Mouilleron, V., Mahamdi, M., Antoun, W., and Seddah, D. Beyond dataset creation: critical view of annotation variation and bias probing of a dataset for online radical content detection. In *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 8640–8663, 2025.

Song, L., Liu, J., Zhang, J., Zhang, S., Luo, A., Wang, S., Wu, Q., and Wang, C. Adaptive in-conversation team building for language model agents. *arXiv preprint arXiv:2405.19425*, 2024.

TogetherAI. Togetherai api. `https://www.together.ai/`.

Wan, R., Wang, H., Huang, T.-H., and Gao, J. From noise to nuance: Enriching subjective data annotation through qualitative analysis. In *Proceedings of the Fourth Workshop on Bridging Human-Computer Interaction and Natural Language Processing (HCI+ NLP)*, pp. 240–254, 2025.

West, A., Weng, Y., Zhu, M., Lin, Z., Ning, Z., and Zhang, Y. Abduct, act, predict: Scaffolding causal inference for automated failure attribution in multi-agent systems. *arXiv preprint arXiv:2509.10401*, 2025.

Wu, Q., Bansal, G., Zhang, J., Wu, Y., Li, B., Zhu, E., Jiang, L., Zhang, X., Zhang, S., Liu, J., et al. Autogen: Enabling next-gen llm applications via multi-agent conversations. In *First Conference on Language Modeling*, 2024.

Xu, Y., Derricks, V., Earl, A., and Jurgens, D. Modeling annotator disagreement with demographic-aware experts and synthetic perspectives. *arXiv preprint arXiv:2508.02853*, 2025.

Yang, A., Li, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Gao, C., Huang, C., Lv, C., et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Ye, R., Tang, S., Ge, R., Du, Y., Yin, Z., Chen, S., and Shao, J. Mas-gpt: Training llms to build llm-based multi-agent systems. *arXiv preprint arXiv:2503.03686*, 2025.

Yoran, O., Amouyal, S. J., Malaviya, C., Bogin, B., Press, O., and Berant, J. Assistantbench: Can web agents solve realistic and time-consuming tasks? *arXiv preprint arXiv:2407.15711*, 2024.

Zhang, G., Wang, J., Chen, J., Zhou, W., Wang, K., and Yan, S. Agentracer: Who is inducing failure in the llm agentic systems? *arXiv preprint arXiv:2509.03312*, 2025a.

Zhang, H., Shi, Y., Gu, X., You, H., Zhang, Z., Gan, L., Yuan, Y., and Huang, J. Graphtracer: Graph-guided failure tracing in llm agents for robust multi-turn deep search. *arXiv preprint arXiv:2510.10581*, 2025b.

Zhang, J., Xiang, J., Yu, Z., Teng, F., Chen, X., Chen, J., Zhuge, M., Cheng, X., Hong, S., Wang, J., et al. Aflow: Automating agentic workflow generation. *arXiv preprint arXiv:2410.10762*, 2024.

Zhang, S., Yin, M., Zhang, J., Liu, J., Han, Z., Zhang, J., Li, B., Wang, C., Wang, H., Chen, Y., et al. Which agent causes task failures and when? on automated failure attribution of llm multi-agent systems. In *Forty-second International Conference on Machine Learning*.

A. Use of Generative AI Tools

Generative AI tools are used solely for minor editing purposes during the manuscript preparation, such as improving sentence fluency and correcting typographical errors. They are not used to generate new content, analyses, or experimental results.

B. Ethics Statement

This dataset is constructed exclusively using publicly available tasks and publicly accessible multi-agent systems. It does not contain any personally identifiable information. To the best of our knowledge, the dataset does not raise any ethical concerns.

C. Annotation Quality and Effort

Dataset Composition. MP-BENCH consists of 169 hand-crafted MAS executions (average 33 interaction steps) and 120 automatically generated executions (average 8 steps), totaling 289 instances with triple expert annotation.

Annotator Qualifications. All annotators possess substantial technical expertise in AI and multi-agent systems: Annotator 1 (7 years in AI engineering and NLP), Annotator 2 (10+ years in applied AI and LLM-based agentic systems), and Annotator 3 (5 years in AI/ML development, LLMs, and AI agents). This ensures annotations reflect genuine expert judgment on MAS failure patterns.

Annotation Effort. Each hand-crafted execution requires approximately 31 minutes per annotator on average, while automatic executions requires approximately 14 minutes, reflecting the complexity difference in execution length and interaction patterns. Across all 289 instances and three annotators, the total annotation effort amounts to approximately 346 expert-hours (115 hours per annotator). Annotators were compensated at competitive rates commensurate with their expertise levels.

This substantial time investment prioritizes annotation quality and multi-perspective coverage over dataset scale, ensuring the reasoning quality essential for reliable evaluation of failure attribution systems. While this approach limits immediate scalability, it establishes a gold-standard benchmark for validating future automated annotation methods.

D. Implementation Details

D.1. Failure Attribution Systems Details

We implement failure attribution systems using a diverse set of LLMs, including both proprietary and open-source models: GPT-4.1 (OpenAI, 2025b), GPT-5.1 (OpenAI, 2025c), o3-mini (OpenAI, 2025d), GPT-oss-120B (Agarwal et al., 2025), Claude-Sonnet-4.5 (OpenAI, 2025a), and Qwen3-8B (Yang et al., 2025). For failure attribution, we adopt the All-at-Once method (Zhang et al.) in a zero-shot setting, which provides a favorable balance between efficiency and accuracy. Unless otherwise specified, we set the number of LLM runs to $N = 3$, the sampling temperature to $\tau = 1.0$, and the maximum output length to 4,196 tokens. Proprietary models are accessed via their official APIs (OpenAI; Anthropic). For the open-source GPT-oss-120B, we use the Together.ai API (TogetherAI). Qwen3-8B is run locally on our own machine equipped with an NVIDIA RTX A6000 GPU.

D.2. Evaluation Details

For failure attribution evaluation, we report nDCG under multiple settings, including nDCG@5 and nDCG@full, with both exponential and linear gain variants. This design accounts for the fact that exponential gain places greater emphasis on steps with high consensus rates. By additionally reporting full-ranking and linear-gain metrics, we provide a more comprehensive evaluation that also reflects performance on lower-consensus, perspective-dependent failure steps. For attribution reasoning evaluation, we use GPT-5.1 as the consolidation and judge model with $\tau = 0.0$ to ensure stable and deterministic judgments. Based on manual inspection, we find that the judge model produces reliable evaluation results.

E. Additional Experimental Results

E.1. Sensitivity Analyses on LLM choice for Consolidation and LLM-as-a-Judge

In our main experiments, we use GPT-5.1 (OpenAI, 2025c) for both annotation consolidation (Section 4.1.4) and LLM-as-a-Judge evaluation (Section 5.2). To verify that our evaluation is robust to these choices, we conduct comprehensive sensitivity analyses using GPT-4.1 (OpenAI, 2025b) and Claude-Sonnet-4.5 (OpenAI, 2025a) as alternative models. With three consolidation models and three judge models, we evaluate all nine possible combinations.

Absolute Score Variation across Judges. Tables 5 and 6 present attribution reasoning scores across all nine configurations for hand-crafted and automatic MAS executions, respectively. We observe that GPT-4.1 as judge consistently assigns higher scores than GPT-5.1 and Claude-Sonnet-4.5. Given that GPT-5.1 and Claude-Sonnet-4.5 possess stronger reasoning capabilities, they likely evaluate more rigorously, resulting in more conservative scoring. Nonetheless, scores across all configurations consistently exceed 7.0, indicating high-quality attribution reasoning from all evaluated systems regardless of judge choice.

Ranking Consistency: Hand-Crafted MAS. Despite vari-

ation in absolute scores, model rankings remain highly consistent. For hand-crafted MAS executions, we compute Kendall's rank correlation coefficient across all $\binom{9}{2} = 36$ configuration pairs. The results demonstrate strong consistency:

- Average $\tau = 0.896$
- Minimum $\tau = 0.733$, Maximum $\tau = 1.00$
- 31 of 36 pairs achieve statistical significance ($p < 0.05$)
- All 36 pairs achieve $p < 0.1$

These correlations substantially exceed conventional thresholds for strong agreement, confirming that model rankings are robust to evaluation configuration.

Ranking Consistency: Automatic MAS. For automatic MAS executions, we observe similarly strong consistency:

- Average $\tau = 0.904$
- Minimum $\tau = 0.733$, Maximum $\tau = 1.00$
- 30 of 36 pairs achieve $p < 0.05$
- All 36 pairs achieve $p < 0.1$

Conclusion. While absolute scores vary with judge model choice—reflecting differences in evaluation stringency—relative rankings remain highly stable across all configurations. This consistency demonstrates that our conclusions about model performance are not artifacts of specific consolidation or judge choices, but reflect genuine differences in failure attribution quality. The strong agreement (average $\tau \simeq 0.90$ for both settings) provides confidence that practitioners can reliably identify top-performing systems regardless of evaluation setup.

E.2. Scaling N in Multi-LLM Failure Attribution

We investigate whether the scaling behavior with respect to N observed in Section 6.2.1 also holds in the multi-LLM failure attribution setting. For multi-LLM configurations, we follow the same setup as in Section 6.2.2, where three LLMs collaboratively perform failure attribution. We vary N from 1 to 10, corresponding to a total of 3 to 30 LLM runs ($3 \times N$). Performance is evaluated using the average nDCG@5 with exponential gain across both hand-crafted and automatically generated MAS datasets. As shown in Figure 5, increasing the number of samples (N) leads to steady performance improvements, consistent with prior observations.

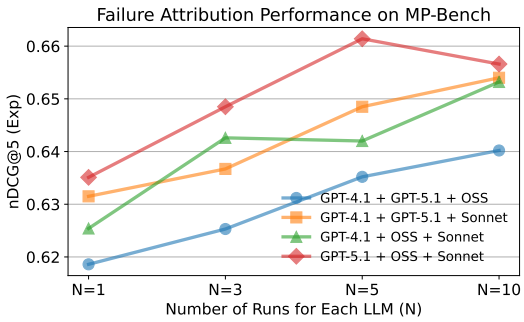

Figure 5. Failure attribution performance across varying numbers of LLM runs (N) on MP-BENCH. OSS denotes GPT-OSS-120B, and Sonnet denotes the Claude-Sonnet-4.5 model.

Table 5. Performance comparison across consolidation and judge model combinations for hand-crafted MAS executions. Sonnet denotes Claude-Sonnet-4.5. Values in parentheses indicate model rankings.

	Consolidation: GPT-4.1			Consolidation: GPT-5.1			Consolidation: Sonnet		
	Judge: GPT-4.1	Judge: GPT-5.1	Judge: Sonnet	Judge: GPT-4.1	Judge: GPT-5.1	Judge: Sonnet	Judge: GPT-4.1	Judge: GPT-5.1	Judge: Sonnet
GPT-4.1	8.49 (3)	7.52 (3)	7.07 (2)	8.60 (3)	7.63 (2)	7.20 (2)	8.48 (3)	7.44 (3)	7.14 (2)
GPT-5.1	8.57 (2)	7.53 (2)	7.04 (3)	8.68 (2)	7.62 (3)	7.14 (4)	8.56 (2)	7.49 (2)	7.03 (3)
O3-mini	8.22 (5)	7.31 (5)	6.93 (5)	8.32 (5)	7.48 (5)	7.06 (5)	8.13 (5)	7.23 (5)	6.92 (5)
Claude-Sonnet-4.5	8.70 (1)	7.82 (1)	7.27 (1)	8.83 (1)	7.95 (1)	7.50 (1)	8.64 (1)	7.71 (1)	7.32 (1)
Qwen3-8B	7.45 (6)	5.97 (6)	5.82 (6)	7.54 (6)	6.11 (6)	5.96 (6)	7.37 (6)	5.92 (6)	5.77 (6)
GPT-oss-120B	8.37 (4)	7.40 (4)	7.03 (4)	8.46 (4)	7.51 (4)	7.18 (3)	8.27 (4)	7.34 (4)	6.97 (4)

Table 6. Performance comparison across consolidation and judge model combinations for automatic MAS executions. Sonnet denotes Claude-Sonnet-4.5. Values in parentheses indicate model rankings.

	Consolidation: GPT-4.1			Consolidation: GPT-5.1			Consolidation: Sonnet		
	Judge: GPT-4.1	Judge: GPT-5.1	Judge: Sonnet	Judge: GPT-4.1	Judge: GPT-5.1	Judge: Sonnet	Judge: GPT-4.1	Judge: GPT-5.1	Judge: Sonnet
GPT-4.1	8.5 (4)	7.66 (4)	7.15 (4)	8.69 (3)	7.78 (4)	7.38 (4)	8.43 (4)	7.49 (4)	7.13 (4)
GPT-5.1	8.84 (1)	8.06 (1)	7.45 (2)	9.01 (2)	8.25 (1)	7.68 (2)	8.81 (1)	8.03 (1)	7.48 (1)
O3-mini	8.13 (5)	7.21 (5)	6.86 (5)	8.28 (5)	7.37 (5)	7.06 (5)	8.03 (5)	7.16 (5)	6.75 (5)
Claude-Sonnet-4.5	8.8 (2)	8.01 (2)	7.5 (1)	9.02 (1)	8.23 (2)	7.73 (1)	8.77 (2)	7.94 (2)	7.47 (2)
Qwen3-8B	7.54 (6)	6.38 (6)	6.03 (6)	7.74 (6)	6.54 (6)	6.24 (6)	7.43 (6)	6.17 (6)	5.93 (6)
GPT-OSS-120B	8.57 (3)	7.87 (3)	7.25 (3)	8.66 (4)	7.92 (3)	7.46 (3)	8.5 (3)	7.66 (3)	7.25 (3)

Brief Description

Annotators will receive json files containing MAS execution logs. Each log includes:

- A task description (e.g., ″Find popular hiking trails with waterfalls in Yosemite″)
- Step-by-step agent actions (e.g., WebSurfer, Verifier, Orchestrator outputs)
- A final system answer (which may or may not be correct)

In the json file, we have multiple fields: ″question″, ″ground_truth″, ″is_corrected″, and ″history.″

- ″question″
  - Describes the user′s task.
- ″ground_truth″
  - Represents the correct answer to the user′s task.
  - If not provided, the annotator must come up with the ground truth and fill it in.
  - If already provided, the annotator does not need to generate it.
- ″is_corrected″
  - indicates whether this MAS execution produced the correct answer.
  - true → The execution produced the correct answer.
  - false → The execution produced an incorrect conclusion.
- ″history″

  Contains a sequence of agent actions, with the following subfields:
  - ″content″: the action taken by each agent.
  - ″step″: The order of agent actions,
  - ″failure_annotation″, ″fail_category″, ″fail_reason″, and ″ideal_action″ are the annotation targets. The freelancer should complete these for each instance.

Detailed Annotation Guides

1. Failure Annotation – Identify whether each agent action was correct (0) or incorrect (1). This field has binary values, 1 or 0. If the same type of failure occurs multiple times, please annotate all such occurrences without omission.

2. Failure Reason – For incorrect actions, briefly describe the reason for failure in clear, natural language.

3. Ideal Action – Suggest the ideal action the agent should have taken. Do not include the actual simulation results of the agent. Instead, provide guidance for the failed action (e.g., ″The WebSurfer agent should have scrolled down the page to explore the information instead of visiting other pages″). The corrected action must be directly aligned with the failure reason you annotate.

4. Ground Truth – The log can optionally contain the ground truth answer for the user query. If it is not provided, please come up with the ground truth.

5. Is Corrected – Judge the correctness of the final response from the MAS with respect to the ground truth answer: “true” if the final answer is correct, “false” if incorrect. If the provided value is wrong, please correct it based on your judgment.

Figure 6. Annotation description provided to annotators.

You are an AI assistant tasked with analyzing a multi-agent conversation history when solving a real-world problem. The problem is: {problem}.
Identify all agents who made errors, at which steps each error occurred, and analyze the nature of these errors.
Here's the conversation:

{chat history content}

Based on this conversation, please provide a detailed analysis of the mistakes in the following JSON format. The output must be a list of objects, ordered by the step number. Each object must include:
1. agent_name: The name of the agents who made any mistakes during the conversation. Directly output the name of the Expert.
2. step_number: The integer step number where the mistake occurred (e.g., 0, 1, 2...). For example, in a conversation structured as follows:
{
"agent a": "xx",
"agent b": "xxxx",
"agent c": "xxxxx",
"agent a": "xxxxxxx"
},

the textual chat history is structured as follows:

Step 0: agent a: x
Step 1: agent b: xxxx
Step 2: agent c: xxxxx
Step 3: agent a: xxxxxxx

each entry represents a 'step' where an agent provides input. The 'x' symbolizes the speech of each agent. Please determine the step numbers where the mistakes occurred.
Please specify the step number as presented."
3. failure_reason: Briefly describe the reason for failure in clear, natural language.
4. ideal_action: Suggest the ideal action the agent should have taken. Do not include actual simulation results; instead, provide guidance aligned with the failure reason (e.g., 'The WebSurfer agent should have scrolled down the page to explore the information instead of visiting other pages').
Please answer ONLY in the following JSON format:
[
{
"agent_name": "AgentName1",
"step_number": "only integer",
"failure_reason": "Description of why it failed",
"ideal_action": "Guidance on what should have been done",
},
...
]
Please order the results by the step number. Do not include any other text or comments.

Figure 7. Prompt for failure attribution method (All At Once (Zhang et al.)).

You are an expert evaluator assessing the quality of a large language model's failure attribution reasoning by comparing it with a human expert annotator's annotation.
Your goal is NOT to check for exact wording matches, but to evaluate whether the model provides a plausible, well-grounded reasoning that aligns with the human annotator's judgment.
—
Task: {question}
Failure Step: {step_num}
Execution Log (context around step {step_num}):
{chat history content}
—
Model Prediction:
- Failure Reason: {pred_fail_reason}
- Ideal Action: {pred_ideal_action}

Human Expert Annotation:
- Failure Reason: {GT_fail_reason}
- Ideal Action: {GT_ideal_action}

—
Evaluation Criteria:
Evaluate the model's prediction by jointly considering the failure reason and ideal action, based on the following aspects:
1. Reasoning Alignment
Does the model capture the core reasoning behind why the step is considered a failure and what should have been done instead, as reflected in the human annotation?

2. Faithfulness to Execution Context
Is the model's reasoning grounded in the actual execution context of this step?
- Does it rely only on information available in the execution trace?
- Does it avoid introducing unsupported assumptions or hindsight knowledge?

3. Coverage and Completeness
Does the model address the key issues and considerations raised by the annotator, even if expressed differently?

4. Plausibility of Ideal Action
Is the proposed ideal action reasonable, actionable, and consistent with the annotator's intent?

—
Scoring Guidelines (1–10 scale):
- 9–10 (Excellent):
Strong alignment with the annotator's reasoning; well-grounded, faithful to the execution context, and captures the key rationale and ideal action clearly.

- 7–8 (Good):
Largely aligned with minor omissions or differences; reasoning is plausible and context-faithful.

- 5–6 (Moderate):
Partially aligned but missing important aspects or containing some questionable assumptions.

- 3–4 (Poor):
Weak alignment; reasoning diverges significantly from the annotator or lacks grounding in the execution context.

- 1–2 (Very Poor):
Largely incorrect, unfaithful to the execution context, or fails to provide a meaningful explanation.

—
Only return the JSON object. Do not include any additional text. Output Format (JSON only):
{ "overall_score": "integer between 1 and 10",
"fail_reason_score": "integer between 1 and 10",
"ideal_action_score": "integer between 1 and 10",
"reasoning": "Concise explanation of the judgment" }

Figure 8. LLM-as-a-Judge prompt for attribution reasoning evaluation.