VERDICT: VERIFICATION-BASED DIAGNOSIS VIA CAUSAL TOPOLOGY FOR MULTI-AGENT FAILURE ATTRIBUTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Figuring out which agent caused a multi-agent system (MAS) to fail and exactly when it happened is still a major challenge. Current methods usually ask an LLM to review execution logs and guess the culprit, but they don't provide estimates of the model's confidence. We developed VERDICT (**VER**ification-based **DI**agnosis via **C**ausal **T**opology), a new framework that brings structured reasoning to this problem. Instead of trying to learn causal relationships from scratch, VERDICT extracts a formal model directly from the system's execution protocol. It introduces two main ideas: first, it leverages the system's known design to construct a causal graph, thereby avoiding the need for causal discovery from massive amounts of data; and second, it uses Halpern–Pearl (HP)-inspired blame diagnostics with bootstrap stability intervals to provide reproducible diagnoses with explicit uncertainty metadata. Extensive experiments demonstrate that VERDICT achieves competitive results without incurring any LLM calls. Notably, on the Who&When benchmark, VERDICT significantly outperforms the LLM-judge baselines released with the benchmark, raising step-level accuracy from 16.5% to 29.3% and agent-level accuracy from 53.2% to 64.1%. These gains are achieved with reproducible causal diagnostics in less than 40 seconds on a single CPU.

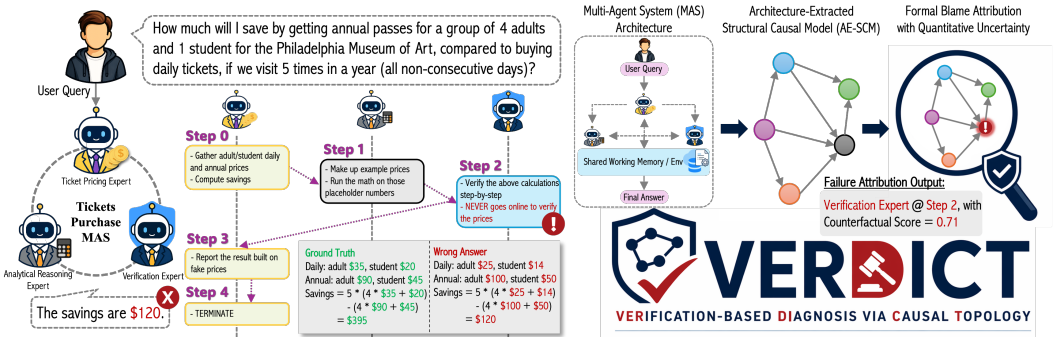


Figure 1: Conceptual illustration of VERDICT: failure attribution in multi-agent systems via causal topology using a ticket pricing example.

1 INTRODUCTION

LLM-powered multi-agent systems (MAS) are increasingly deployed in enterprise workflows, from autonomous coding assistants (Hong et al., 2024) to complex web navigation (Zhou et al., 2024) and general-purpose task solving (Mialon et al., 2024). When multiple cooperating agents fail to complete a task, identifying the responsible agent and the precise step at which the failure originated is essential for debugging and for building reliable agentic AI. Yet, this attribution problem has no straightforward solution.

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.

1.1 WHY IS MULTI-AGENT FAILURE ATTRIBUTION HARD?

Root cause analysis in MAS faces a fundamental structural challenge that departs from traditional software engineering. In traditional systems, debugging is guided by observable error signals, such as stack traces, exception logs, and assertion failures, that pinpoint where execution deviated from its intended path. However, MAS often lack these localized faults. Each agent in a MAS typically operates on a partial view of the global task, constrained by its specific context window. Consequently, an agent may select an action that is locally rational yet globally incorrect. When multiple agents interact sequentially, the impact of locally plausible decisions can compound, leading to an erroneous final outcome without triggering any functional error flags. This phenomenon creates a "logical silent failure" that is effectively invisible to standard diagnostics. Classical fault localization uses spectrum-based methods (Jones & Harrold, 2005; Abreu et al., 2007) or anomaly detection (Chandola et al., 2009) to correlate failures with observable signals, but MAS logs often show only a sequence of "completed" statuses, offering no traditional evidence of where the logical divergence began.

Consider the example in Figure 1. A user asks a Tickets Purchase MAS to compute the savings of annual passes versus daily tickets for a group of four adults and one student over five visits to the Philadelphia Museum of Art. The Ticket Pricing Expert is tasked with gathering current adult and student rates, the Analytical Reasoning Expert with running the arithmetic, and the Verification Expert with checking the result. The system returns $120, while the ground-truth answer is $395. It remains unclear whether the Ticket Pricing Expert failed to retrieve the actual prices, the Analytical Reasoning Expert fabricated placeholder values and computed on them, or the Verification Expert audited the arithmetic step-by-step without ever cross-checking the prices online, as the lack of intermediate error signals leaves the candidates indistinguishable to standard debugging tools.

1.2 FROM INFERENCE TO VERIFICATION

Existing approaches to failure attribution treat it as an inference problem from execution logs. The state-of-the-art methods (Zhang et al., 2025b) feed execution traces to a large language model, asking it to identify the responsible agent and decisive step from the text. The agent-level accuracy (All-at-Once) and step-level accuracy (Step-by-Step) are both low, reflecting the fundamental difficulty of failure attribution with text patterns alone. When prompted to predict what would have happened under an alternative action, LLMs may produce narratives that are internally coherent but do not correspond to the actual causal mechanism (Zheng et al., 2023). An alternative is to apply causal discovery algorithms that learn causal structure from execution traces. Such methods, however, require large annotated datasets with non-trivial human-labelling overhead. Supervised fine-tuning does not close this gap: Yu et al. (2025) report that a fine-tuned Qwen-2.5-7B judge has an extremely low (3.5%) step-exact accuracy.

We propose a new third paradigm, **VER**ification-based **DI**agnosis via **C**ausal **T**opology (VERDICT). The key observation is that multi-agent execution already has a well-defined causal structure, one specified by the system architecture rather than hidden in the data. The State $\rightarrow$ Action $\rightarrow$ Environment $\rightarrow$ NextState chain is a definitional property of how turn-based multi-agent systems operate. VERDICT reads this structure directly from the system's design specification and uses it as a formal scaffold for counterfactual reasoning. Within this framework, VERDICT computes counterfactual scores using a deterministic, embedding-based approximation (Section 3.2) that combines the causal structure with task-level diagnostic inputs (an expected output and/or a failure symptom, when available)—matching regression-test and benchmark debugging settings rather than fully unsupervised attribution. Throughout, "verification" denotes checkable reproducibility: every diagnostic object VERDICT emits is a deterministic function of the trace and the declared architecture that a third party can re-derive exactly, not a formal verification of causal claims.

1.3 SIGNIFICANCE AND CONTRIBUTIONS

VERDICT yields capabilities that inference-based methods cannot provide, regardless of the quality of their underlying models. When an LLM-based inference method attributes a failure to a particular agent or step, the attribution is a point prediction: the method offers no signal of its own reliability. VERDICT instead computes attribution through a sequence of formally defined operations: structural equation instantiation from the system architecture, composite counterfactual score computation, and Halpern-Pearl-inspired sufficiency/minimality diagnostics (Halpern, 2015). Every intermediate

result is a well-defined mathematical object that can be independently inspected and reproduced. The framework further produces bootstrap stability intervals over candidate-correction proxy outcomes, providing an explicit signal that distinguishes stable attributions from ambiguous ones.

Contributions:

- **Architecture-extracted structural causal models.** We formalize multi-agent execution semantics as an Architecture-Extracted SCM (AE-SCM) and show that the causal graph needed for counterfactual attribution can be derived directly from the system's execution protocol. This eliminates the sample-complexity barrier of causal discovery.
- **HP-inspired blame diagnostics with stability metadata.** We adapt Halpern–Pearl's actual causality theory (Halpern, 2015) to develop HP-inspired blame diagnostics that incorporate bootstrap stability intervals over candidate-correction proxy outcomes.

2 Problem Formulation

2.1 Multi-Agent Execution Model

Many LLM MAS, including MetaGPT (Hong et al., 2024), CAMEL (Li et al., 2023), ReAct (Yao et al., 2023), and WebArena (Zhou et al., 2024), expose a turn-based execution trace in which agents observe state, act, and update a shared environment. We model such a system as

$$\mathcal{A} \;=\; \langle\, \mathcal{N},\, \mathcal{S},\, \mathcal{E},\, \mathcal{C},\, \phi \,\rangle, \tag{1}$$

where $\mathcal{N} = \{1, \ldots, n\}$ is the agent set, $\mathcal{S} = \prod_{i \in \mathcal{N}} \mathcal{S}^i$ the joint state space, $\mathcal{E}$ the shared environment, $\mathcal{C} \in \{0,1\}^{n \times n}$ the communication matrix ($\mathcal{C}_{ij} = 1$ iff agent j receives information from agent i), and $\phi : \{0, \ldots, T-1\} \to \mathcal{N}$ the scheduler. The outcome is $Y \in \{0, 1\}$, with 0 denoting success and 1 denoting failure. Because $\mathcal{C}$ and ϕ come from the MAS design, they can be converted directly into the AE-SCM in Section 3.1.

Lowercase symbols denote realized values of the corresponding SCM variables. We write $s_t = (s_t^1, \ldots, s_t^n)$ for the joint state, a_t for the action taken by agent $\phi(t)$, and e_t for the environment response. A trajectory of length T is

$$\tau \;=\; \big(s_0,\, a_0,\, e_0,\, s_1,\, a_1,\, e_1,\, \ldots,\, s_{T-1},\, a_{T-1},\, e_{T-1},\, s_T,\, y\big), \tag{2}$$

where $y \in \{0, 1\}$ is the realized outcome. Failure attribution is constrained to this trace and the architecture in Eq. (1). The core difficulty is local observability: at step t, the active agent $\phi(t)$ chooses action a_t using only its local context $s_t^{\phi(t)}$, so an action that appears reasonable locally may still lead to global failure.

2.2 Decisive Error

To connect the observed trajectory to failure attribution, we first define a counterfactual target: whether changing that action would have flipped the final outcome from failure to success.

Counterfactual notation. Uppercase A_t is the action random variable, lowercase a_t its realized value in τ. The intervention $\mathrm{do}(A_t := \tilde{a}_t)$ fixes the pre-t history, sets $A_t{=}\tilde{a}_t$, and regenerates post-t variables via the structural equations of Section 3.1; $\tau^{\,\mathrm{do}(A_t:=\tilde{a}_t)}$ denotes the counterfactual trajectory and $y(\tau)$ its outcome.

Definition 1 (Decisive Error)**.** *Given a failed trajectory τ with outcome $y(\tau) = 1$, a tuple (i, t, a_t), where agent $i = \phi(t)$ takes action a_t at step t, is a **decisive error** if there exists an alternative action $\tilde{a}_t \neq a_t$ such that the counterfactual trajectory would have succeeded:*

$$y\big(\tau^{\,\mathrm{do}(A_t:=\tilde{a}_t)}\big) = 0. \tag{3}$$

Since LLM logs cannot execute arbitrary $\tilde{a}_t$, VERDICT estimates this target using the architecture-derived scaffold and deterministic semantic counterfactual scores over the candidates in Eq. (11). When multiple decisive errors are plausible, we report the primary one by score ranking.

Definition 2 (Primary Decisive Error Estimate)**.** *Let* $\mathrm{cf}(t)$ *be the composite counterfactual score (defined in Eq.* (16)*).* VERDICT *selects the primary decisive-error estimate as*

$$t^* = \min\{t : \mathrm{cf}(t) = \max_{t'} \mathrm{cf}(t')\},$$

where the earliest-step rule resolves ties by temporal precedence, matching the minimality (used in Eq. (18)*). The attributed agent is* $\phi(t^*)$*.*

Problem 1 (Failure Attribution)**.** *Given a failed trajectory* τ *with* $y(\tau) = 1$*, output four diagnostic objects: the attributed pair* $(t^*, \phi(t^*))$*, the ranked score landscape* $\{\mathrm{cf}(t)\}_{t=0}^{T-1}$ *over all candidate steps, an HP-style sufficiency flag indicating whether the top step is clearly separated from the rest of the trace, and a bootstrap stability interval for the selected step.*

Remark 1. VERDICT *does not assume uniqueness of decisive errors; it selects a primary candidate by score ranking with temporal tie-breaking. Applicability scope is detailed in Appendix B.*

3 VERDICT METHODOLOGY

Section 2 defines the ideal decisive error, but exact counterfactual resimulation is unavailable for LLM agents. VERDICT therefore turns the ideal target into a computable three-phase pipeline, as shown in Figure 2.

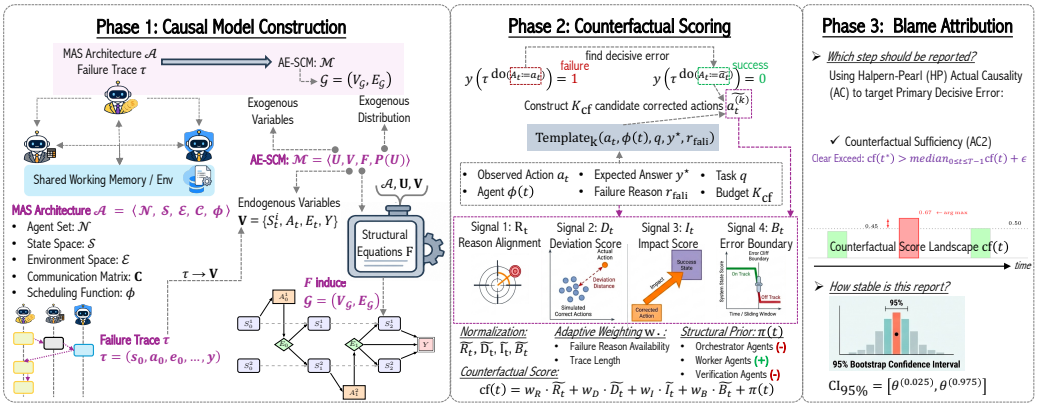


Figure 2: End-to-end overview of the VERDICT pipeline, with detailed illustration of Phases I–III.

3.1 CAUSAL MODEL CONSTRUCTION

VERDICT reads the agent set, communication matrix, and scheduling rule from a MAS to build a causal model.

Definition 3 (Architecture-Extracted SCM (AE-SCM))**.** *Given a multi-agent system* $\mathcal{A} = \langle \mathcal{N}, \mathcal{S}, \mathcal{E}, \mathcal{C}, \phi \rangle$ *and a trajectory* τ *of length* T*, the AE-SCM is a tuple* $\mathcal{M} = \langle \mathbf{U}, \mathbf{V}, \mathbf{F}, P(\mathbf{U}) \rangle$ *where:*

Exogenous variables.

$$\mathbf{U} = \{U_0\} \cup \{U_t^A : 0 \le t \le T-1\} \cup \{U_t^E : 0 \le t \le T-1\}. \tag{4}$$

U_0 encodes the task specification and initial user intent; U_t^A captures LLM stochasticity at step t; U_t^E represents environmental noise (API latency, network conditions) at step t.

Endogenous variables.

$$\mathbf{V} = \{S_t^i : 0 \le t \le T,\ i \in \mathcal{N}\} \cup \{A_t, E_t : 0 \le t \le T-1\} \cup \{Y\}. \tag{5}$$

S_t^i is the message and prompt history visible to agent i at step t (its context window); A_t is the action taken at step t by the scheduled agent $\phi(t)$; E_t is the environment response at step t; and $Y \in \{0, 1\}$ is the binary task outcome introduced in Section 2.

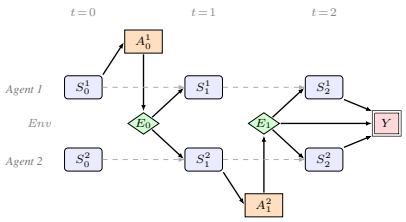


Figure 3: Toy AE-SCM for a two-agent, two-action trace with scheduling $\phi(0) = 1$, $\phi(1) = 2$. Purple boxes are states S_t^i, orange boxes are actions A_t, green diamonds are environment responses E_t, and the double-lined node is outcome Y. Solid arrows follow Eqs. (7)–(10); dashed arrows show state persistence. Exogenous noise nodes are omitted for clarity.

Structural equations $\mathbf{F}$*:*

$$S_0^i = f_{init}(U_0,\, i) \tag{6}$$

$$A_t = f_A\big(S_t^{\phi(t)},\, U_t^A\big) \tag{7}$$

$$E_t = f_E\big(A_t,\, E_{t-1},\, U_t^E\big) \tag{8}$$

$$S_{t+1}^i = f_S\big(S_t^i,\, A_t,\, E_t,\, \mathcal{C}_{\phi(t),i},\, \mathbf{1}[i = \phi(t)]\big) \tag{9}$$

$$Y = f_Y\big(\{S_T^i\}_{i\in\mathcal{N}},\, E_{T-1},\, U_0\big) \tag{10}$$

Eqs. (6)–(10) *form a time-unrolled causal story: Eq.* (6) *initializes each agent's context; Eq.* (7) *is the action mechanism (the variable later intervened on as* $\mathrm{do}(A_t := \tilde{a}_t)$ *in Definition 1); Eq.* (8) *updates the shared environment with* $E_{-1} \equiv e_{init}$*; Eq.* (9) *propagates the action into next contexts gated by* $\mathcal{C}_{\phi(t),i}$ *and* $\mathbf{1}[i = \phi(t)]$*; Eq.* (10) *maps terminal states to the outcome. Together these identify downstream variables affected by an intervention on* A_t*. The functions* $f_{init}, f_A, f_E, f_S, f_Y$ *need not admit closed forms: they denote the update rules realized by the LLM agents and the environment, while the SCM fixes which variables each function reads and writes.*

Exogenous distribution. $P(\mathbf{U})$ *is a prior over the exogenous noise.*

Induced causal graph. The structural equations $\mathbf{F}$ induce the causal topology $\mathcal{G} = (V_{\mathcal{G}}, E_{\mathcal{G}})$ used in Figure 2: we set $V_{\mathcal{G}} = \mathbf{V}$ and add an edge from each endogenous parent variable to the variable it directly determines in Eqs. (6)–(10). Because all dependencies follow the time-unrolled order, $\mathcal{G}$ is a directed acyclic graph (DAG) derived from $\mathbf{F}$, not an additional learned model.

Architecture specification tiers. We distinguish two specification patterns: *(i) Fully-specified architectures*, where roles and topology are fixed at design time (e.g., Magentic-One (Fourney et al., 2024), MetaGPT (Hong et al., 2024)) and the AE-SCM is built once; and *(ii) Topology-template architectures*, where topology is design-time but agents are instantiated per trace from its metadata (e.g., CaptainAgent (Song et al., 2024)).

Algorithm 1 in Appendix C (complexity $O(nT)$) gives the pseudo-code and Figure 3 shows a toy 2-agent example.

3.2 Counterfactual Scoring

Phase 2 implements deterministic semantic counterfactual scoring. For each step t, VERDICT compares the observed action a_t against a candidate set $\widetilde{\mathcal{A}}_t = \{\tilde{a}_t^{(k)}\}_{k=1}^{K_{\mathrm{cf}}}$ and assigns a composite score $\mathrm{cf}(t)$ that combines four signals: alignment with the failure reason, deviation from corrected alternatives, alignment of corrections with the expected answer, and on/off-track transition. Throughout, $y^\star$ denotes the expected-answer text (not the outcome variable Y).

Counterfactual candidate construction. For each observed action a_t, VERDICT constructs K_{cf} candidates via fixed string templates over the task description q, the expected answer $y^\star$, the failure-reason string r_{fail}, the scheduled agent $\phi(t)$ (rendered as its trace-recorded role/name), and a_t:

$$\tilde{a}_t^{(k)} = \mathrm{Template}_k\big(a_t, \phi(t), q, y^\star, r_{\mathrm{fail}}\big), \qquad k = 1, \ldots, K_{\mathrm{cf}}. \tag{11}$$

The implementation details are provided in Algorithm 2; the configuration evaluated in this paper uses $y^\star$, matching regression-test and benchmark debugging settings. The set $\widetilde{\mathcal{A}}_t$ is a deterministic,

reproducible substitute for the unknown successful alternative $\tilde{a}_t$ of Definition 1: the same inputs always produce the same candidates, eliminating both LLM-call cost and the variance an LLM-based repair generator would introduce.

Composite counterfactual scoring. VERDICT approximates the unobservable counterfactual via four semantic signals over the sentence-embedding map $\mathrm{emb}(\cdot)$ (`all-MiniLM-L6-v2`; Section 4). When r_{fail} is absent (e.g., TRAIL-GAIA), it is replaced by a smart auto-reason, a content-anchored surrogate reason constructed deterministically from the trace itself (Section 4.1); if no reason is available, we set $R_t{=}B_t{=}0$.

Signal 1: Reason Alignment (R_t). This signal asks whether the observed action at step t semantically matches the stated or automatically generated failure reason:

$$R_t = \begin{cases} \cos\big(\mathrm{emb}(a_t),\, \mathrm{emb}(r_{\text{fail}})\big), & \text{if } r_{\text{fail}} \text{ is available,} \\ 0, & \text{otherwise.} \end{cases} \tag{12}$$

High R_t indicates the action text resembles the failure symptom.

Signal 2: Deviation Score (D_t). This signal asks how far the observed action is from the deterministic corrected actions in $\widetilde{\mathcal{A}}_t$:

$$D_t = 1 - \frac{1}{K_{\text{cf}}} \sum_{k=1}^{K_{\text{cf}}} \cos\big(\mathrm{emb}(a_t),\, \mathrm{emb}(\tilde{a}_t^{(k)})\big). \tag{13}$$

A high D_t marks a step where observed behavior diverges from plausible repairs in $\widetilde{\mathcal{A}}_t$.

Signal 3: Impact Score (I_t). This signal asks whether the corrected actions would point toward the expected answer:

$$I_t = \frac{1}{K_{\text{cf}}} \sum_{k=1}^{K_{\text{cf}}} \cos\big(\mathrm{emb}(\tilde{a}_t^{(k)}),\, \mathrm{emb}(y^\star)\big). \tag{14}$$

This separates repairs that merely differ from the observed action from repairs that move toward the expected successful outcome.

Signal 4: Error Boundary (B_t). Whether t lies near a low-to-high failure-alignment transition. With the windowed mean $\overline{R}_{[\ell,u]} = \frac{1}{u-\ell+1}\sum_{j=\ell}^{u} R_j$, half-window $w{=}\max(5, \lfloor T/6 \rfloor)$, and trace-clipped endpoints $\ell_t^- {=} \max(0, t{-}w)$, $u_t^+ {=} \min(T{-}1, t{+}w)$, define

$$B_t = \overline{R}_{[t,u_t^+]} - \overline{R}_{[\ell_t^-,t]}. \tag{15}$$

A large positive B_t marks an on-track-to-off-track transition.

Counterfactual score aggregation. The four raw signals are rank-normalized to $[0,1]$ and combined with a trace-dependent weight schedule:

$$\mathrm{cf}(t) = w_R \cdot \tilde{R}_t + w_D \cdot \tilde{D}_t + w_I \cdot \tilde{I}_t + w_B \cdot \tilde{B}_t + \pi(t), \tag{16}$$

where $\tilde{\cdot}$ denotes rank-normalized values; $\mathrm{cf}(t)$ is the computable proxy for Eq. (3), from which Phase 3 selects t^*. The schedule (w_R, w_D, w_I, w_B) obeys two structural properties from the signal definitions: when r_{fail} is unavailable, Eq. (12) forces $R_t{=}0$, so $w_R{=}0$ and its mass is reallocated to w_D, w_I; when r_{fail} is available, R_t is the most localizing cue on long traces, so w_R grows with T. Specific values used in all experiments are calibrated against the `all-MiniLM-L6-v2` similarity distribution and reported as a fixed configuration in Section 4 (Table 1); sensitivity is in Appendix C.3. The additive prior $\pi(t) \in [-0.10, +0.05]$ penalizes orchestrator agents (frequent relayers) and verification agents (inheritors of upstream alignment) and boosts worker agents; values come from architectural roles, not tuning, and removing π degrades Who&When agent accuracy by 6.0 pp.

3.3 BLAME ATTRIBUTION

Phase 3 answers which step should be reported, and how stable is that report? The phase has three outputs: an HP-style sufficiency/minimality diagnostic, a bootstrap stability interval, and a deterministic causal-chain narrative.

HP-inspired blame diagnostics. The Halpern-Pearl (HP) Actual Causality (AC) definition (Halpern, 2015) requires three criteria for $\{A_t = a_t\}$ to be an actual cause of $\{Y = 1\}$: actual occurrence (AC1), counterfactual sufficiency (AC2), and minimality (AC3). We take the modified definition as the reference point because it fixes the contingency set at its actually observed values, the natural reading for a single-trace post-mortem. Since natural-language LLM mechanisms are not executable, VERDICT uses the HP conditions as a diagnostic structure over the semantic score landscape rather than as literal formal verification.

AC1: actual occurrence. Both $\{A_t = a_t\}$ and $\{Y = 1\}$ hold by construction, since VERDICT only processes failed traces.

AC2: sufficiency proxy. Since VERDICT cannot execute the literal HP intervention (which requires an existential witness set $\mathbf{W} \subseteq \mathbf{V} \setminus \{A_t, Y\}$ flipping the outcome), it tests whether the selected step is clearly separated from the score landscape:

$$\mathrm{cf}(t^*) > \underset{0 \le t \le T-1}{\mathrm{median}}\, \mathrm{cf}(t) + \epsilon, \qquad \epsilon = 0.05. \tag{17}$$

A step failing this predicate is labeled HP-ambiguous and routed to human/LLM review. Eq. (17) is an operational sufficiency diagnostic, not a literal proof of HP's existential AC2.

AC3: minimality. Among steps tied at the maximum composite score, VERDICT selects the earliest:

$$t^* = \min\big\{t : \mathrm{cf}(t) = \max_{t'} \mathrm{cf}(t')\big\}. \tag{18}$$

Each candidate cause is a singleton action assignment and is therefore minimal by construction, so Eq. (18) is a deterministic reporting convention (earlier errors can enable later propagated ones), not a uniqueness claim.

Bootstrap stability intervals. While the HP flag tests landscape separation, the bootstrap interval asks whether the candidate corrections at t^* are stable. VERDICT converts each $\tilde{a}_t^{(k)}$ into a per-candidate impact $I_{t,k} = \cos(\mathrm{emb}(\tilde{a}_t^{(k)}), \mathrm{emb}(y^\star))$ (so I_t in Eq. (14) equals $\frac{1}{K_{\mathrm{cf}}} \sum_k I_{t,k}$) and a binary semantic-success proxy $o_{t,k} = \mathbf{1}[I_{t,k} \ge \delta]$ with δ=0.35; $o_{t,k}$ is a deterministic proxy (not the true outcome). For step t^*, draw N_{boot} resamples of size K_{cf} with replacement from $(o_{t^*,k})_{k=1}^{K_{\mathrm{cf}}}$, with means $\hat{\theta}_b = \frac{1}{K_{\mathrm{cf}}} \sum_k o_{t^*,k}^{(b)}$ for $b = 1, \ldots, N_{\mathrm{boot}}$; with $\theta^{(q)}$ the empirical q-quantile of $\{\hat{\theta}_b\}_{b=1}^{N_{\mathrm{boot}}}$,

$$\mathrm{CI}_{95\%} = \big[\theta^{(0.025)}, \theta^{(0.975)}\big]. \tag{19}$$

Because the bootstrap resamples deterministic template outcomes rather than an i.i.d. sample, it measures sensitivity to reweighting the candidate bank at t^*, never a calibrated posterior of correctness.

Causal chain construction. Finally, VERDICT converts the formal diagnosis into a contrastive human-readable narrative covering trace initiation, the decisive step, downstream propagation, and outcome (Miller, 2019). Because narratives are derived deterministically from the causal graph and score landscape, they are fully reproducible.

4 EXPERIMENTAL RESULTS

Benchmarks. We evaluate VERDICT on two benchmarks. The **Who&When** (Zhang et al., 2025b) benchmark has two subsets: *Algorithm-Generated* traces from CaptainAgent (Song et al., 2024), short and dynamic (n=126, mean 6.3 steps, 86 distinct expert types); and *Hand-Crafted* traces from Magentic-One (Fourney et al., 2024), long hub-and-spoke traces with five fixed agents (n=58, mean 50.1 steps, range 3–128). Each trace has expert-annotated ground truth (responsible agent, decisive step, natural-language failure reason). The **TRAIL-GAIA** (Deshpande et al., 2025) benchmark is a span-level attribution benchmark on GAIA (Mialon et al., 2024) traces under the OpenDeepResearch architecture (n=113 of 117; 4 traces skipped).

Baselines and metrics. We compare VERDICT against Tier-1 methods in Appendix A, using Agent-Level Accuracy and Step-Level Accuracy (Exact Step and Exact Step$\pm$1). Step$\pm$1 grants a one-step margin of error for correct attribution.

Implementation. VERDICT uses `all-MiniLM-L6-v2` (Reimers & Gurevych, 2019) embeddings with $K_{\text{cf}} = 10$, $\delta = 0.35$, $\epsilon = 0.05$, fixed seed 42, and the weight schedule of Table 1 (held fixed across both benchmarks; sensitivity in Appendix C.3). Bootstrap stability intervals use $N_{\text{boot}} = 1{,}000$ resamples per attribution and $N_{\text{boot}} = 10{,}000$ for the aggregate 184-trace Who&When CIs. The pipeline runs in 39.5 s on Who&When (0.21 s/trace) and 28.8 s on TRAIL-GAIA (0.25 s/trace) with *zero LLM API calls*, 7.6–21.6× faster than our local GPU runs of the All-at-Once and A2P judges with `Qwen-2.5-7B`.

Table 1: Fixed weight schedule for Eq. (16), used identically across Who&When and TRAIL-GAIA with no per-trace or per-benchmark retuning. The no-reason row is structural: Eq. (12) forces R_t=0, so w_R=0 and the freed mass goes to w_D, w_I. Other values are calibrated once against the `all-MiniLM-L6-v2` similarity distribution; sensitivity sweep in Appendix C.3.

Reason available	Trace length T	w_R	w_D	w_I	w_B
no	any	0.00	0.45	0.45	0.10
yes	$T \leq 15$	0.35	0.20	0.25	0.20
yes	$15 < T \leq 40$	0.40	0.10	0.20	0.30
yes	$T > 40$	0.45	0.10	0.15	0.30

4.1 RESULTS ON WHO&WHEN

Main Results. Table 2 shows that VERDICT improves over the three published Zhang et al. LLM-judge baselines on all reported aggregate metrics: +10.9 pp agent accuracy, +12.8 pp step-exact accuracy, and +27.2 pp Step±1. Aggregate bootstrap 95% CIs over the 184 traces are $[57.1, 70.7]$ for agent accuracy, $[22.8, 35.9]$ for step-exact accuracy, and $[45.7, 60.3]$ for Step±1, with lower bounds above the corresponding Zhang baseline point estimates. The VERDICT row also consumes the annotated failure reason, which the LLM baselines do not receive; the annotation-free auto-reason variant below removes this input. Every non-LLM baseline we evaluated, including similarity-only ranking by $\cos(\text{emb}(a_t), \text{emb}(r_{\text{fail}}))$, the role prior alone, and a majority-class agent predictor, stays at or below 57.6% agent and 22.8% step accuracy overall (similarity-only vs. VERDICT: McNemar p=0.0009 and p=0.024).

A2P with `gpt-oss-120b` reports higher overall step accuracy, and FAMAS leads on Hand-Crafted step localization. The comparison instead highlights a cost–reproducibility frontier. To separate A2P's prompting scaffold from model scale, we reran the same A2P prompt with `Qwen-2.5-7B-Instruct`. Under full-denominator evaluation, A2P-Qwen reaches 50.8%/32.5% agent/step on Algorithm-Generated and 36.2%/12.1% on Hand-Crafted, while VERDICT reaches 60.3%/35.7% and 72.4%/15.5%, respectively (overall agent gap: McNemar p=0.0002). Compared to FAMAS, VERDICT wins on agent accuracy across both subsets and on Algorithm-Generated step accuracy, while FAMAS leads on Hand-Crafted step at the cost of trace re-executions and many LLM calls.

Comparison with frontier LLM judges. An independent evaluation by Yu et al. (2025) stress-tests eleven LLM-judge backbones on Who&When. On the Hand-Crafted subset, every tested LLM judge remains below 10% step-exact, while VERDICT reaches 15.5% with zero LLM calls. The full model-by-model comparison is reported in Appendix C.2.

VERDICT as a First-Stage Retriever. Beyond single-step prediction, VERDICT's counterfactual scores form a ranked list over candidate steps. The ground-truth decisive step appears in the top-5 for 77.2% of all Who&When traces (92.1% Algorithm-Generated, 44.8% Hand-Crafted) and in the top-10 for 88.0% overall (100.0% Algorithm-Generated, 62.1% Hand-Crafted); on the ∼50-step Hand-Crafted traces, the ground-truth step's mean rank of 12.84 shrinks the review set roughly fourfold.

Annotation Dependence: Discriminability Analysis. The smart auto-reason variant achieves 56.5% agent accuracy, 25.5% step accuracy, and 48.4% Step±1, reducing dependence on human-written failure reasons while still outperforming the published Zhang et al. LLM-judge baselines on all three metrics. Here we report the underlying *discriminability* statistic that explains this recovery.

Table 2: Comparison on the Who&When benchmark. Overall Agent/Step values are weighted by the stated subset sizes (126 Algorithm-Generated, 58 Hand-Crafted); Step±1 uses the published aggregate when subset Step±1 is not reported. Best values are bolded only among methods with the corresponding metric reported. VERDICT's main advantage is the zero-call deterministic cost profile, diagnostic outputs, and top-k retrieval behavior, not universal top-1 step superiority.

	Alg-Generated		**Hand-Crafted**		**Overall**			**LLM**
Method	Agent	Step	Agent	Step	Agent	Step	Step±1	calls
Published LLM-based baselines (Zhang et al., 2025b)								
`All-at-Once`	52.7%	13.0%	54.3%	4.4%	53.2%	10.3%	21.3%	1
`Step-by-Step`	30.6%	20.4%	33.6%	7.9%	31.5%	16.5%	25.5%	many
`Binary Search`	37.1%	20.3%	44.0%	6.9%	39.3%	16.1%	18.1%	many
VERDICT (ours)	60.3%	35.7%	**72.4%**	15.5%	**64.1%**	29.3%	**52.7%**	**0**
Δ vs. best Zhang et al. baseline	+7.6pp	+15.3pp	+18.1pp	+7.6pp	+10.9pp	+12.8pp	+27.2pp	—
A2P (West et al., 2025)								
`gpt-oss-120b`	**65.4%**	**47.5%**	58.6%	29.3%	63.3%	**41.7%**	—	1
`Qwen-2.5-7B`	50.8%	32.5%	36.2%	12.1%	46.2%	26.1%	—	1
FAMAS (Ge et al., 2025)	55.6%	23.8%	62.1%	**41.4%**	57.6%	29.3%	—	many

Let g be the ground-truth decisive step and let $\mathcal{T}_{-g} = \{t : 0 \le t < T,\, t \ne g\}$. For a failure reason r, define

$$\Delta_{\text{disc}}(r) = \cos\big(\text{emb}(r),\, \text{emb}(a_g)\big) - \frac{1}{|\mathcal{T}_{-g}|} \sum_{t \in \mathcal{T}_{-g}} \cos\big(\text{emb}(r),\, \text{emb}(a_t)\big). \tag{20}$$

Smart auto-reasons and human reasons are statistically indistinguishable on this measure ($+0.065$ vs. $+0.065$; $p = 0.98$, Cohen's $d = -0.002$), and both are significantly more discriminative than generic reasons ($+0.011$; $p < 0.0001$, $d = 0.43$). Content-anchored auto-reasons therefore provide the same step-level specificity as expert annotations.

4.2 RESULTS ON TRAIL-GAIA

To test whether VERDICT transfers, we evaluate on TRAIL-GAIA using the smart auto-reason generator (Section 4.1) because TRAIL does not provide natural-language failure reasons. All other weights, priors, and thresholds are identical to Who&When. VERDICT achieves 28.3% location accuracy under the `smart_reason` configuration, ranking 4th of 9 systems and beating five of eight published LLM judges on the matched TRAIL-GAIA Location Accuracy metric. With the same `smart_reason` configuration, the top-5 hit rate is 59.3%. On HIGH-impact errors, VERDICT reaches 54.2% step-exact, suggesting that architecture-grounded signals are most useful when the localized step is load-bearing for the final failure. Full TRAIL-GAIA tables are reported in Appendix C.5.

5 CONCLUSION

We introduced VERDICT, a deterministic, architecture-guided framework for multi-agent failure attribution. Instead of relying on opaque LLM inference or data-heavy learning, VERDICT uses the system's own architecture to build causal models and compute deterministic semantic counterfactual scores. Its three-phase pipeline, covering SCM construction, counterfactual scoring, and Halpern-Pearl blame attribution, achieves competitive results on the Who&When benchmark. Remarkably, it achieves these results in under 40 seconds on the CPU without requiring any expensive LLM calls, and it transfers to TRAIL-GAIA with unchanged weights, priors, and thresholds. VERDICT also provides unique diagnostic tools like reproducible causal chains that any researcher can verify. Ultimately, by shifting failure diagnosis from LLM inference to deterministic computation, VERDICT provides the transparency and reliability essential for real-world production environments.

AI USE STATEMENT

We used generative AI tools to assist in editing the text, enhancing figure aesthetics, and providing draft critique. All content was reviewed and edited by the authors. The authors take responsibility for the final text, claims, and artifacts.

REFERENCES

Rui Abreu, Peter Zoeteweij, and Arjan J. C. van Gemund. On the accuracy of spectrum-based fault localization. In *Testing: Academic and Industrial Conference Practice and Research Techniques*, pp. 89–98. IEEE, 2007.

Liliana Ardissono, Luca Console, Anna Goy, Giovanna Petrone, Claudia Picardi, Marino Segnan, and Daniele Theseider Dupré. Cooperative model-based diagnosis of web services. In *16th International Workshop on Principles of Diagnosis (DX-05)*, pp. 125–130, 2005.

Mert Cemri, Melissa Z. Pan, Shuyi Yang, Lakshya A. Agrawal, Bhavya Chopra, Rishabh Tiwari, Kurt Keutzer, Aditya Parameswaran, Dan Klein, Kannan Ramchandran, Matei Zaharia, Joseph E. Gonzalez, and Ion Stoica. Why do multi-agent LLM systems fail? *Advances in Neural Information Processing Systems 38 (NeurIPS) Datasets and Benchmarks Track*, 2025.

Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Computing Surveys (CSUR)*, 41(3):1–58, 2009.

Hana Chockler and Joseph Y. Halpern. Responsibility and blame: A structural-model approach. *Journal of Artificial Intelligence Research*, 22:93–115, 2004.

Darshan Deshpande, Varun Gangal, Hersh Mehta, Jitin Krishnan, Anand Kannappan, and Rebecca Qian. TRAIL: Trace reasoning and agentic issue localization. *arXiv preprint arXiv:2505.08638*, 2025.

Will Epperson, Gagan Bansal, Victor Dibia, Adam Fourney, Jack Gerrits, Erkang Zhu, and Saleema Amershi. Interactive debugging and steering of multi-agent AI systems. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, 2025. doi: 10.1145/3706598.3713581.

Adam Fourney, Gagan Bansal, Hussein Mozannar, Cheng Tan, Eduardo Salinas, Friederike Niedtner, Grace Proebsting, Griffin Bassman, Jack Gerrits, Jacob Alber, et al. Magentic-One: A generalist multi-agent system for solving complex tasks, 2024.

Yu Ge, Linna Xie, Zhong Li, Yu Pei, and Tian Zhang. Who is introducing the failure? automatically attributing failures of multi-agent systems via spectrum analysis. *arXiv preprint arXiv:2509.13782*, 2025.

Mingyang Geng, Shanzhi Gu, Zhipeng Liu, Chuanfu Xu, Zhaoyang Qu, and Haotian Wang. Failure localization in multi-agent code generation via knowledge-guided and transferable reasoning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(1):318–326, 2026. doi: 10.1609/aaai.v40i1.36993.

Joseph Y. Halpern. A modification of the Halpern-Pearl definition of causality. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 3022–3033, 2015.

Joseph Y. Halpern. *Actual Causality*. MIT Press, 2016.

Joseph Y. Halpern and Judea Pearl. Causes and explanations: A structural-model approach. Part I: Causes. *British Journal for the Philosophy of Science*, 56(4):843–887, 2005.

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

Azam Ikram, Sarthak Chakraborty, Subrata Mitra, Shiv Saini, Saurabh Bagchi, and Murat Kocaoglu. Root cause analysis of failures in microservices through causal discovery. In *Advances in Neural Information Processing Systems*, pp. 31158–31170, 2022.

James A. Jones and Mary Jean Harrold. Empirical evaluation of the Tarantula automatic fault-localization technique. In *Proceedings of the 20th IEEE/ACM International Conference on Automated Software Engineering*, pp. 273–282, 2005.

David Lewis. Causation. *The Journal of Philosophy*, 70(17):556–567, 1973.

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

Mingjie Li, Zeyan Li, Kanglin Yin, Xiaohui Nie, Wenchi Zhang, Kaixin Sui, and Dan Pei. Causal Inference-Based Root Cause Analysis for Online Service Systems with Intervention Recognition. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 3230–3240. ACM, 2022.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, 2024.

Ming Ma, Jue Zhang, Fangkai Yang, Yu Kang, Qingwei Lin, Saravan Rajmohan, and Dongmei Zhang. DoVer: Intervention-driven auto debugging for LLM multi-agent systems. *arXiv preprint arXiv:2512.06749*, 2025.

Yuan Meng, Shenglin Zhang, Yongqian Sun, Ruru Zhang, Zhilong Hu, Yiyin Zhang, Chenyang Jia, Zhaogang Wang, and Dan Pei. Localizing Failure Root Causes in a Microservice through Causality Inference. In *2020 IEEE/ACM 28th International Symposium on Quality of Service (IWQoS)*, pp. 1–10. IEEE, 2020.

Grégoire Mialon, Clémentine Fourrier, Craig Swift, Thomas Wolf, Yann LeCun, and Thomas Scialom. GAIA: A benchmark for general AI assistants. In *International Conference on Learning Representations*, 2024.

Tim Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 267:1–38, 2019.

Judea Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2nd edition, 2009.

Judea Pearl, Madelyn Glymour, and Nicholas P. Jewell. *Causal Inference in Statistics: A Primer*. John Wiley & Sons, 2016.

Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, pp. 3982–3992, 2019.

Linxin Song, Jiale Liu, Jieyu Zhang, Shaokun Zhang, Angela Luo, Shuaicheng Wang, Qingyun Wu, and Chi Wang. Adaptive in-conversation team building for language model agents, 2024.

Kai Sun, Wenqiang Li, Bo Dong, Yuxin Lin, Jingyao Zhang, and Bin Shi. Scope delineation before localization: A two-stage framework for enhancing failure attribution in multi-agent systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 33108–33116, 2026.

Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. Solving math word problems with process- and outcome-based feedback. *arXiv preprint arXiv:2211.14275*, 2022.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-Shepherd: Verify and reinforce LLMs step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9426–9439. Association for Computational Linguistics, 2024.

Yawen Wang, Wenjie Wu, Junjie Wang, and Qing Wang. From flat logs to causal graphs: Hierarchical failure attribution for LLM-based multi-agent systems. *arXiv preprint arXiv:2602.23701*, 2026.

Alva West, Yixuan Weng, Minjun Zhu, Zhen Lin, Zhiyuan Ning, and Yue Zhang. Abduct, Act, Predict: Scaffolding causal inference for automated failure attribution in multi-agent systems. *arXiv preprint arXiv:2509.10401*, 2025.

W. Eric Wong, Ruizhi Gao, Yihao Li, Rui Abreu, and Franz Wotawa. A survey on software fault localization. *IEEE Transactions on Software Engineering*, 42(8):707–740, 2016.

James Woodward. *Making Things Happen: A Theory of Causal Explanation*. Oxford University Press, 2003.

Li Wu, Johan Tordsson, Erik Elmroth, and Odej Kao. MicroRCA: Root Cause Localization of Performance Issues in Microservices. In *NOMS 2020 – IEEE/IFIP Network Operations and Management Symposium*, pp. 1–9. IEEE, 2020.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023.

Yifan Yu, Moyan Li, Shaoyuan Xu, Jinmiao Fu, Xinhai Hou, Fan Lai, and Bryan Wang. Correct: Condensed error recognition via knowledge transfer in multi-agent systems. *arXiv preprint arXiv:2509.24088*, 2025.

Guibin Zhang, Junhao Wang, Junjie Chen, Wangchunshu Zhou, Kun Wang, and Shuicheng Yan. AgenTracer: Who is inducing failure in the LLM agentic systems? *arXiv preprint arXiv:2509.03312*, 2025a.

Shaokun Zhang, Ming Yin, Jieyu Zhang, Jiale Liu, Zhiguang Han, Jingyang Zhang, Beibin Li, Chi Wang, Huazheng Wang, Yiran Chen, and Qingyun Wu. Which agent causes task failures and when? on automated failure attribution of LLM multi-agent systems. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267, pp. 76583–76599. PMLR, 2025b.

Chujie Zheng, Zhenru Zhang, Beichen Zhang, Runji Lin, Keming Lu, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. ProcessBench: Identifying process errors in mathematical reasoning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1009–1024. Association for Computational Linguistics, 2025.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Advances in Neural Information Processing Systems*, 2023.

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Yonatan Bisk, Daniel Fried, Uri Alon, et al. WebArena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations*, 2024.

Chenyang Zhu, Spencer Hong, Jingyu Wu, Kushal Chawla, Yuhui Tang, Youbing Yin, Nathan Wolfe, Erin Babinsky, and Daben Liu. RAFFLES: Reasoning-based attribution of faults for LLM systems. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7659–7688, 2026.

A Related Work: A Larger Picture and Comparable Solutions

Failure attribution in LLM multi-agent systems intersects with several well-established research areas, including actual causality, root cause analysis in distributed systems, classical software fault localization, and process reward modeling for reasoning traces. Each of these fields offers methods that might initially seem like potential baselines. Nevertheless, we must distinguish between *methodological relatedness* and *evaluation compatibility*: although two methods may share mathematical or conceptual roots, they operate on fundamentally different input types, task definitions, output spaces, or data conditions. To clarify the scope of this paper's empirical claims—and to specify which existing works we compare against and which we do not—we categorize the relevant literature into three tiers.

A.1 Tier 1: Directly Comparable Baselines

In this tier, methods address the same problem of predicting which agent and which step led to a failed multi-agent trajectory. They utilize the same benchmark-compatible execution traces and produce the same (agent, step) predictions required by the Who&When and TRAIL-GAIA evaluation protocols. On the first benchmark, these methods include the three LLM-judge prompting regimes from the Who&When paper (Zhang et al., 2025b) (All-at-Once, Step-by-Step, and Binary Search), A2P Scaffolding (West et al., 2025), and FAMAS (Ge et al., 2025). On the second benchmark, the methods include the eight LLM judges tabulated by TRAIL (Deshpande et al., 2025). We benchmark VERDICT against every Tier-1 method whose predictions are available on the respective evaluation surface (Table 2 for Who&When, Table 6 for TRAIL-GAIA).

Note that Tier 1 also includes other methods such as AgenTracer (Zhang et al., 2025a), FLKR (Geng et al., 2026), and CHIEF (Wang et al., 2026). Nevertheless, we exclude them from the comparison due to significant barriers. AgenTracer's replay-based training infrastructure is not publicly released. FLKR (Geng et al., 2026) only reports agent-level accuracy on a code-generation variant. The code for CHIEF was not released at the time of our evaluation, which prevents faithful replication of the Who&When/TRAIL-GAIA trace serializations. Concurrent methods SDBL (Sun et al., 2026) and RAFFLES (Zhu et al., 2026) also report agent-and-step attribution on Who&When but, like A2P, rely on multi-stage LLM reasoning over every trace.

A.2 Tier 2: Methodologically Adjacent but Evaluation-Incompatible

Methods in this tier share conceptual or algorithmic ancestry but cannot be compared with VERDICT without altering their task definition, required inputs, or output semantics. The incompatibility in each case is structural.

Process Reward Models (PRMs) (Uesato et al., 2022; Lightman et al., 2024; Wang et al., 2024; Zheng et al., 2025) operate on single-agent mathematical-reasoning traces with per-step correctness labels, not on multi-agent failed trajectories.

Spectrum-based fault localization (SBFL) methods such as Tarantula (Jones & Harrold, 2005; Abreu et al., 2007; Wong et al., 2016) require per-statement coverage spectra from a pass/fail-labeled test suite, a signal definitionally absent from LLM-agent execution logs.

Distributed-systems root cause analysis methods, including CIRCA (Li et al., 2022), MicroRCA (Wu et al., 2020), MicroCause (Meng et al., 2020), and the intervention-based approach of Ikram et al. (2022), operate on time-series metric streams (latency, error rate, CPU) with numerically defined anomalies. But they have no mechanism for ingesting natural-language messages or tool calls.

DoVer (Ma et al., 2025) reorients evaluation toward outcome-flip and progress metrics measured across repeated trials, a philosophically distinct target from decisive-step attribution on a single trajectory. It cannot be benchmarked on the Who&When/TRAIL evaluation surface without a significant structural change.

AGDebugger (Epperson et al., 2025) is an interactive debugging UI rather than an automated predictor; it does not emit a machine-comparable (agent, step) output.

MAST (Cemri et al., 2025) labels each trajectory with one of 14 trace-level failure-mode categories rather than per-agent per-step attributions, so MAST-Data is not suitable as an attribution benchmark.

Classical model-based and cooperative diagnosis, including diagnosis of composite web services (Ardissono et al., 2005), anticipates our setting of distributed components jointly failing a task, but presupposes formal component behavior models that natural-language LLM agents lack.

A.3 Tier 3: Theoretical and Philosophical Foundations

Pearl's Structural Causal Model framework (Pearl, 2009; Pearl et al., 2016), the Halpern-Pearl definition of actual causality (Halpern & Pearl, 2005; Halpern, 2015; 2016), Chockler and Halpern's degree-of-responsibility theory (Chockler & Halpern, 2004), and the broader causal-explanation literature (Lewis, 1973; Woodward, 2003; Miller, 2019) are the mathematical and conceptual foundation on which VERDICT is built. They are not empirical estimators with benchmarkable outputs.

B Scope of Applicability

VERDICT applies to any LLM MAS that satisfies four structural conditions:

- (S1) *Turn-based scheduling.* At each discrete step t, a single agent takes an action, so actions can be enumerated as a finite sequence $a_0, a_1, \ldots, a_{T-1}$. The scheduling of agent actions may be deterministic (e.g., round-robin), planned (e.g., by an orchestrator), or adaptive (e.g., by a router), but each step must have a single attributable active agent.
- (S2) *Known communication topology.* The communication matrix $\mathcal{C}$, or a parametric template from which $\mathcal{C}$ is instantiated per trace, is available at design time. This is what allows the AE-SCM to be extracted from architecture rather than learned from data.
- (S3) *Observable message trace.* Every agent action a_t and environment response e_t is recorded in the execution log with sufficient fidelity to compute an embedding. Tool calls, retrieval queries, and code output are included while internal chain-of-thought may be omitted.
- (S4) *Point-localizable failure mode.* The final failure admits at least one decisive error, i.e., there exists a step whose counterfactual correction would have restored success. This excludes failures where the root cause is the absence of a capability (e.g., missing a verification agent) or a systemic workflow defect.

Failure attribution for any LLM MAS outside the above scope is considered future research.

C Extended Evaluation Details

Algorithm 1: Phase 1: AE-SCM Instantiation

Input: Architecture spec $\mathcal{A} = \langle \mathcal{N}, \mathcal{S}, \mathcal{E}, \mathcal{C}, \phi \rangle$; execution trace τ of length T
Output: Instantiated AE-SCM $\mathcal{M} = \langle \mathbf{U}, \mathbf{V}, \mathbf{F}, P(\mathbf{U}) \rangle$

$\mathcal{N}, \mathcal{C}, \phi \leftarrow$ ResolveFromSpec$(\mathcal{A}, \tau)$ `// given prior (Tier 1) or instantiated per trace (Tier 2)`
$\mathbf{U} \leftarrow \{U_0\}$; $\mathbf{V} \leftarrow \emptyset$
for $t = 0$ **to** $T - 1$ **do**
  for *each* $i \in \mathcal{N}$ **do**
    $\mathbf{V} \leftarrow \mathbf{V} \cup \{S_t^i\}$
  end
  $\mathbf{V} \leftarrow \mathbf{V} \cup \{A_t, E_t\}$
  $\mathbf{U} \leftarrow \mathbf{U} \cup \{U_t^A, U_t^E\}$
end
$\mathbf{V} \leftarrow \mathbf{V} \cup \{S_T^i : i \in \mathcal{N}\} \cup \{Y\}$
$\mathbf{F} \leftarrow$ InstantiateStructEq$(\mathcal{N}, \mathcal{C}, \phi)$ `// Eqs. (6)–(10)`
$P(\mathbf{U}) \leftarrow$ PriorOver$(\mathbf{U})$
BindObservedValues$(\mathbf{V}, \tau)$ `// fill` S_t^i, A_t, E_t, Y `from the trace`
return $\mathcal{M} = \langle \mathbf{U}, \mathbf{V}, \mathbf{F}, P(\mathbf{U}) \rangle$

Algorithm 2: Deterministic construction of semantic counterfactual candidates.

Input: Observed action a_t, scheduled agent $\phi(t)$, task q, expected answer $y^\star$, failure reason r_{fail}, budget K_{cf}
Output: Candidate set $\widetilde{\mathcal{A}}_t$

$C \leftarrow [\,]$
if $y^\star$ *is available* **then**
  add "agent $\phi(t)$ correctly produces: $y^\star$" to C
  add "Correct response leading to: $y^\star$" to C
end
if r_{fail} *is available* **then**
  add "agent $\phi(t)$ avoids: r_{fail}; correctly handles the task" to C
  add "Instead of: r_{fail}, the correct action succeeds" to C
end
if q *is available* **then**
  add "agent $\phi(t)$ correctly executes: q" to C
end
while $|C| < K_{cf}$ **do**
  add a deterministic rewrite combining a prefix of a_t with the best available target among $y^\star$, r_{fail}, and q
end
return first K_{cf} candidates in C

Table 3: Step-exact accuracy on the Hand-Crafted subset of Who&When for all eleven LLM-judge backbones stress-tested by Yu et al. (2025), plus the random-guessing floor and a fine-tuned variant. Numbers are taken from Table 1 of Yu et al. (2025) (Human-Crafted, Acc@0 column).

Method	**Step Acc.**
Random guessing	$\sim$3.0%
Llama-8b	1.7%
Qwen-2.5-7b	3.5%
Qwen-3-30b	1.7%
Qwen-3-80b	6.9%
DeepSeek-R1	3.5%
Gemini-2.5-flash	5.2%
Gemini-2.5-pro	3.5%
GPT-4o-mini	3.5%
GPT-4o	3.5%
GPT-5-nano	1.7%
GPT-5	8.6%
Qwen-2.5-7B (fine-tuned)	3.5%
VERDICT (ours)	**15.5%**

C.1 COUNTERFACTUAL CANDIDATE CONSTRUCTION ALGORITHM

Algorithm 2 gives the deterministic candidate-construction routine referenced by Eq. (11). The algorithm consumes whichever of $(y^\star, r_{\text{fail}}, q)$ is available and pads to K_{cf} candidates via deterministic rewrites of a_t.

C.2 ADDITIONAL MAIN-TEXT COMPARISONS

Frontier LLM judges. Table 3 reports the model-by-model Hand-Crafted comparison summarized in Section 4.1.

C.3 HYPERPARAMETER SENSITIVITY

Five axes were swept with others held at defaults. The published configuration is a conservative setting: some post-hoc variants improve individual metrics, such as prior scale 2.0 on Who&When, but we retain the default to avoid selecting hyperparameters on the evaluation set (Table 4). Tuning a single global weight vector under leave-one-out cross-validation, so that each trace is scored with weights fit only on the other traces, raises overall agent accuracy to 69.6%.

Table 4: Consolidated hyperparameter sensitivity on Who&When.

Knob	**Default**	**Best non-default**	**Agent (def. / alt.)**	**Gap**
K_{cf} (6 values)	$K_{\text{cf}} = 10$	$K_{\text{cf}} = 5$	64.1% / 62.5%	+1.6 pp
Boundary window (7 formulas)	`max(5, T/6)`	two formulas (tie)	64.1% / 64.1%	0.0 pp
Similarity metric (4 choices)	cosine	Pearson	64.1% / 64.1%	0.0 pp
Embedding backbone (5 models)	`all-MiniLM-L6-v2`	`all-MiniLM-L12-v2`	64.1% / 61.4%	+2.7 pp
Prior scale (7 values)	1.0	2.0	64.1% / 67.4%	−3.3 pp

C.4 VERDICT-UNIQUE OUTPUTS

Triage signals. CI width and the HP-style flag provide diagnostic metadata about the stability of semantic counterfactual samples. The causality margin (gap between top-1 and top-2 scores) is a

finer-grained signal: agent accuracy rises from 59.5% for near-zero margins to 74.1% for margins ≥ 0.20 in Table 5.

C.5 Extended TRAIL-GAIA Analysis

The headline TRAIL-GAIA comparison is shown in Table 6.

Table 5: Agent accuracy vs. causality margin on Who&When.

Margin bucket	n	**Agent accuracy**
$[0.00, 0.02)$	37	59.5%
$[0.02, 0.05)$	36	58.3%
$[0.05, 0.10)$	46	65.2%
$[0.10, 0.20)$	38	65.8%
$[0.20, 0.50)$	27	**74.1%**

Table 6: TRAIL-GAIA Location Accuracy (n=113). LLM-judge numbers from Deshpande et al. (2025).

Rank	**System**	**GAIA Loc. Acc.**	**Deterministic**
1	Gemini-2.5-Pro	54.6%	×
2	O3	53.5%	×
3	Gemini-2.5-Flash	37.2%	×
4	**VERDICT (ours)**	**28.3%**	✓
5	Claude-3.7-Sonnet	20.4%	×
6	GPT-4.1	10.7%	×
7	O1	4.0%	×
8	LLAMA-4-Maverick	2.3%	×
9	LLAMA-4-Scout	0.0%	×