EVENTLEDGER: TYPED COUNTERFACTUAL CREDIT FOR MULTI-AGENT LLM ORCHESTRATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Multi-agent LLM systems solve a task by orchestrating typed events: an orchestrator spawns roles, delegates subgoals, issues tool calls, requests critiques, aggregates candidates, and decides to stop. Supervision still arrives once at the end, so the outcome reward is broadcast to every event and a decisive tool call is credited the same as a wasted search or a premature stop. Recent counterfactual credit methods sharpen this signal, but they assume a single homogeneous unit (a message or an agent), are evaluated only with a programmatic verifier, and do not score the stopping decision that governs cost. We present EVENTLEDGER, which treats one execution as an orchestration event graph and assigns each event an interventional, auditable process reward. Typed counterfactual operators define a type-compatible intervention for every event type, including a force-continue/force-stop pair that makes termination a scored decision. A perturb-rollout engine estimates the marginal effect of every replayed event without bias, pairs factual and counterfactual continuations with common random numbers, and spends its replay budget on the highest-leverage events. For tasks without a verifier, a cross-family judge committee is calibrated isotonically on verifiable anchors and abstains by a conformal rule, which bounds the credit bias under an explicit transfer assumption. The rewards are combined with potential-based shaping and distilled into a graph-aware student that makes no API calls at inference. Across coding, math, text-to-SQL and open-ended question answering, an orchestrator trained with EVENTLEDGER rewards reaches an average score of 87.75 against 85.57 for the strongest baseline; the budgeted estimator needs 39.8% of the replay jobs of the full estimator; the oracle pipeline lowers a single judge's expected calibration error from 0.44 to 0.12 at 93.33% retention; and the student reaches 0.80 agreement with the teacher at $0.05\times$ its scoring cost while matching the end-task scores of the teacher-trained orchestrator.

1 INTRODUCTION

A multi-agent LLM system produces not only an answer but also a *process*. An orchestrator spawns a solver and a critic, delegates a subgoal, calls a tool, reads the observation, requests a critique, revises, aggregates, and eventually decides that it is done. Each of these is a different kind of decision with a different cost and a different way of going wrong (Cemri et al., 2026; Zhang et al., 2025), and the last one, when to stop, is the one that most directly sets the bill. Yet the learning signal these systems receive is almost always a single scalar at the end. Outcome reward is broadcast uniformly, so a pivotal schema lookup, a redundant repeated search, and a premature stop all receive the same credit, and a policy trained on that signal cannot tell which of its decisions mattered.

A recent line of work replaces the broadcast with counterfactual credit: replay the transcript with one unit removed or altered and measure how much the outcome moves (Chen et al., 2026; Li et al., 2026; Deshmukh et al., 2026; Tong & Ben-Gal, 2026; Tan et al., 2026). Three restrictions remain. First, the unit is homogeneous (a message or an agent), so a tool call and a critique are intervened on in the same way, although deleting a critique and corrupting a tool argument are not comparable counterfactuals. Second, evaluation is confined to tasks with a programmatic verifier, because the counterfactual continuation needs to be scored. Third, the stopping decision is never scored as an event, although it determines how much of the budget is spent.

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

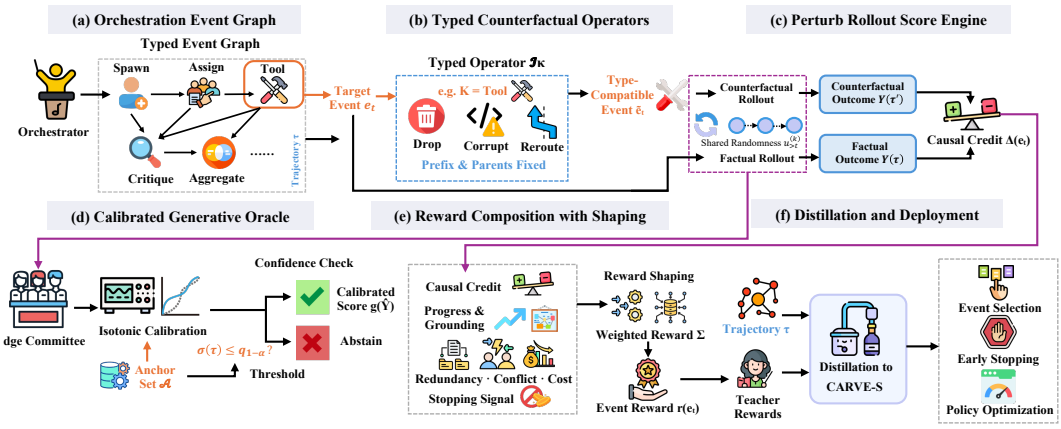

Figure 1: **Overview of EVENTLEDGER.** (a) An execution is an orchestration event graph of typed events with causal parents. (b) A typed operator $\mathcal{I}_\kappa$ produces a type-compatible counterfactual $\tilde{e}_t$ with prefix and parents fixed. (c) Factual and counterfactual continuations share downstream randomness $u_{>t}^{(k)}$, and their outcome difference estimates the causal credit $\Delta(e_t)$. (d) On tasks without a verifier, a cross-family judge committee is calibrated on a verifiable anchor set and abstains when its dispersion exceeds a conformal threshold. (e) Credit is combined with potential-based progress, grounding, redundancy, conflict, cost and stopping terms. (f) The event rewards are distilled into EVENTLEDGER-S, which drives event selection, early stopping and policy optimization at deployment.

EVENTLEDGER (Figure 1) addresses each restriction. It represents an execution as an *orchestration event graph* with typed events and causal parents (Section 3.1); defines a minimal *typed counterfactual operator* for every event type, including a force-continue/force-stop pair for termination (Section 3.2); estimates the interventional effect of each event with a paired, budgeted rollout engine (Section 3.3); replaces the verifier on open-ended tasks with a judge committee that is calibrated on verifiable anchors and abstains when its members disagree (Section 3.4); and distills the resulting rewards into a student that scores events without API calls (Sections 3.5 and 3.6).

The finer signal helps most where coarse ones hurt. Every trained baseline loses accuracy on GSM8K relative to the single-agent GLM-5.2 baseline (95.00): eight of the nine fall to between 84.44 and 90.00 and COSAC slips to 94.44, whereas EVENTLEDGER reaches 95.44. Against the strongest baseline, HCAPO (Tan et al., 2026), the whole +2.18 average margin comes from MBPP and GSM8K (Table 2). The student-trained orchestrator matches the teacher-trained one on every benchmark while using 52.2% fewer deployment tokens than the untrained orchestrator (Table 4).

2 RELATED WORK

Counterfactual credit for LLM agents. Classical multi-agent RL assigns counterfactual credit with a centralized critic (Foerster et al., 2018) or with cooperative-game attribution (Shapley, 1953). For LLM teams, C3 exploits the determinism of cooperative transcripts to compute exact message-level credit (Chen et al., 2026), CCPO optimizes policies with counterfactual agent credit (Li et al., 2026), COSAC derives a critic-free sequential estimator with analytic bias and variance bounds (Deshmukh et al., 2026), CoFi-PGMA studies counterfactual policy gradients under filtered feedback (Tong & Ben-Gal, 2026), HCAPO brings hindsight counterfactual credit to long-horizon agents (Tan et al., 2026), and credit re-assignment has been used for interactive environment generalization (He et al., 2025); a recent survey covers the area (Zhang, 2026a). All of these intervene on one homogeneous unit and are evaluated on verifiable math and code. EVENTLEDGER intervenes on typed orchestration events, scores the stopping decision, and extends to non-verifiable tasks through a calibrated oracle.

Process rewards and failure attribution in multi-agent systems. Process reward models score intermediate steps of single-agent reasoning (Lightman et al., 2024; Wang et al., 2024; Zou et al., 2025) and agent trajectories (Xi et al., 2026). MASPRM trains a multi-agent PRM over partial transcripts and uses it as an inference-time controller (Yazdani et al., 2025), reward modeling has been applied directly to orchestration (Tsang et al., 2026) and RL to orchestration traces (Zhang,

2026b), and MAS-ProVe shows that process verification of multi-agent systems is unreliable when done by a single judge (Venkataramani et al., 2026). Failure-attribution work locates the agent or step responsible for a failure (Cemri et al., 2026; Zhang et al., 2025). EVENTLEDGER differs in deriving its targets from interventions rather than from learned or judged process labels, and in treating credit conservation as an audit rather than a constraint (Section 3.3).

Judges for non-verifiable tasks. Relative judging frameworks such as RULER rank whole trajectories with an LLM judge (Corbitt et al., 2025), and hybrid reward normalization targets process-supervised non-verifiable agentic tasks (Xu et al., 2025). EVENTLEDGER calibrates a cross-family committee against verifiable anchors, abstains with a conformal threshold (Angelopoulos & Bates, 2023), and uses the calibrated judge to score counterfactual continuations rather than to rank trajectories holistically.

Orchestration and shaping. Self-consistency, debate, role-based conversation and graph-structured agents (Wang et al., 2023; Du et al., 2023; Wu et al., 2023; Zhuge et al., 2024) supply our untrained baselines, and potential-based shaping (Ng et al., 1999) underlies our progress term.

3 METHOD

3.1 ORCHESTRATION EVENT GRAPH

One execution is a trajectory $\tau = (G_0, e_1, o_1, G_1, \ldots, e_T, G_T)$, where G_t is the team state after t events and each event e_t has a type $\kappa(e_t) \in \mathcal{K} = \{\text{SPAWN}, \text{ASSIGN}, \text{DELEGATE}, \text{MSG}, \text{TOOL}, \text{OBS}, \text{CRITIQUE}, \text{REVISE}, \text{AGGREGATE}, \text{STOP}\}$ and causal parents $\mathrm{pa}(e_t)$ given by data and control dependencies. Events follow type-specific mechanisms and the outcome is a readout of the terminal state,

$$e_t = f_{\kappa(e_t)}\big(\mathrm{pa}(e_t), u_t\big), \quad u_t \sim P_u, \qquad Y(\tau) = \rho(G_T) \in [0,1], \tag{1}$$

with u_t the exogenous randomness (decoding seeds, tool nondeterminism). Worker roles (solvers, searchers, tool users, critics, verifiers, an aggregator and a stopper) are GLM-5.2 (Lai, 2026) behind an API, differing only in system prompt; the orchestrator issuing the events is either GLM-5.2 (untrained baselines) or a trainable Qwen-3.5 policy (Qwen Team, 2026). On verifiable tasks ρ is a programmatic verifier V; otherwise it is the calibrated oracle of Section 3.4.

Assumption 1 (Restorable prefix)**.** *Given the transcript prefix $s_t \triangleq \tau_{<t}$, the environment carries no hidden state, so the continuation law after any replacement of e_t depends only on $(s_t, \tilde{e}_t)$. Deterministic replay given $(s_t, u_{>t})$ is not required for unbiasedness; it only determines how tightly common random numbers couple a factual and a counterfactual continuation.*

This extends the determinism that C3 exploits for message-level credit (Chen et al., 2026) to the typed event space. Under it, the intervention $\mathrm{do}(e_t \leftarrow \tilde{e}_t)$ has a well-defined post-intervention law that holds s_t fixed and replays downstream mechanisms, $P(\tau' \mid \mathrm{do}(e_t \leftarrow \tilde{e}_t)) = \prod_{j>t} \pi(e_j \mid \mathrm{pa}(e_j))\big|_{e_t=\tilde{e}_t}$.

Definition 2 (Event credit)**.** *The credit of e_t is the interventional contrast between keeping it and replacing it with a type-compatible counterfactual from operator $\mathcal{I}_\kappa$,*

$$\Delta(e_t) = \mathbb{E}_{\tau' \sim P(\cdot \mid \mathrm{do}(e_t \leftarrow e_t))}\big[Y(\tau')\big] - \mathbb{E}_{\tilde{e}_t \sim \mathcal{I}_{\kappa(e_t)}(e_t)}\, \mathbb{E}_{\tau' \sim P(\cdot \mid \mathrm{do}(e_t \leftarrow \tilde{e}_t))}\big[Y(\tau')\big]. \tag{2}$$

The first term re-runs the continuation with the factual event but fresh downstream randomness, rather than using the single realized $Y(\tau)$. This is what makes the estimator below unbiased; crediting against the realized outcome would conflate the effect of the event with the luck of one continuation.

3.2 TYPED COUNTERFACTUAL OPERATORS

TCO gives every event type a minimal operator, a Markov kernel that edits only the target event,

$$\mathcal{I}_\kappa : \mathcal{E}_\kappa \to \Delta\big(\mathcal{E}_\kappa \cup \{\varnothing\}\big), \qquad \mathrm{pa}(e_t),\ s_t \text{ fixed}, \tag{3}$$

with $\varnothing$ the null event. For MSG we delete, nullify or misroute; for TOOL we drop the call, corrupt its arguments or delay its observation; for DELEGATE we reroute or skip; for CRITIQUE we remove or

Table 1: **Typed counterfactual operators.** Every operator edits only the target event; its prefix and parents are held fixed.

Event type	Operators	Question the counterfactual asks
SPAWN, ASSIGN	skip; assign to a different role	did this role need to exist here?
DELEGATE	reroute to another agent; skip	did the recipient matter?
MSG	delete; nullify content; misroute	did the content reach the right agent?
TOOL, OBS	drop call; corrupt arguments; delay observation	did the tool's result matter, and when?
CRITIQUE, REVISE	remove; replace with a generic critique	was the critique specific?
AGGREGATE	drop one source; aggregate before verification	which inputs did the aggregate need?
STOP / non-stop	force-continue (STOP $\mapsto \varnothing$); force-stop ($e_t \mapsto$ STOP)	would more work have helped; was the remaining work needed?

genericize; for AGGREGATE we drop one source or aggregate before verification (Table 1, which also states the question each counterfactual asks). Termination gets a pair of operators,

$$\mathcal{I}_{\text{STOP}}: \quad \underbrace{\text{STOP} \mapsto \varnothing}_{\text{FORCE-CONTINUE}}, \qquad \underbrace{\text{non-stop } e_t \mapsto \text{STOP}}_{\text{FORCE-STOP}}, \tag{4}$$

so that stopping, and failing to stop, both become scored events. Type compatibility is the point of the design: deleting a critique and corrupting a tool argument are different questions about what the event contributed, and a single message-deletion operator answers only one of them.

3.3 PERTURB-ROLLOUT SCORE ENGINE

PRS estimates equation 2 by restoring s_t, drawing $\tilde{e}_t \sim \mathcal{I}_\kappa$, and continuing under the frozen behavior policy; continuations are scored by V on verifiable tasks and by the oracle otherwise. The factual and counterfactual continuations of draw k share the downstream randomness $u^{(k)}$ (common random numbers, CRN),

$$\widehat{\Delta}(e_t) = \frac{1}{K}\sum_{k=1}^{K}\Big[\rho\big(G_T(s_t, e_t, u^{(k)})\big) - \rho\big(G_T(s_t, \tilde{e}_t^{(k)}, u^{(k)})\big)\Big], \qquad \tilde{e}_t^{(k)} \sim \mathcal{I}_\kappa(e_t). \tag{5}$$

Proposition 3 (Unbiasedness and variance reduction)**.** *Under Assumption 1,* $\mathbb{E}[\widehat{\Delta}(e_t)] = \Delta(e_t)$ *for any coupling of the paired continuations. With* $\rho_{\text{fac}}, \rho_{\text{cf}}$ *the paired readouts,*

$$\text{Var}\big(\widehat{\Delta}_{\text{CRN}}\big) = \tfrac{1}{K}\big(\text{Var}\rho_{\text{fac}} + \text{Var}\rho_{\text{cf}} - 2\text{Cov}(\rho_{\text{fac}}, \rho_{\text{cf}})\big) \le \text{Var}\big(\widehat{\Delta}_{\text{ind}}\big) \quad \textit{if } \text{Cov}(\rho_{\text{fac}}, \rho_{\text{cf}}) \ge 0. \tag{6}$$

Unbiasedness needs only that each term is drawn from the correct marginal law; the coupling affects variance alone (proof in Appendix A.1). This matters in practice because API providers do not guarantee seed-exact replay: when the shared seed fails to reproduce a continuation, CRN degrades toward independent sampling, which loses variance reduction but not correctness. To bound cost, a leverage score ranks events and only the top m are replayed,

$$\mathcal{S} = \text{Top-}m\big\{h(e_t)\big\}, \qquad h(e_t) = \widehat{\mathbb{E}}\,|\Delta(e_t)| + \zeta\,\text{Unc}(e_t), \qquad \text{cost} = O(mK), \tag{7}$$

where $\widehat{\mathbb{E}}|\Delta|$ and Unc come from the student of the previous training round (a type prior in the first round). Unselected events receive the credit predicted by the student; this trades bias on low-leverage events for cost, and we measure its effect by replacing the leverage ranking with a random choice of m events (Table 3). Finally we *monitor*, but do not enforce, the efficiency identity of Shapley attribution (Shapley, 1953),

$$\text{gap}(\tau) = \textstyle\sum_t \widehat{\Delta}(e_t) - \big(Y(\tau) - Y_\varnothing\big), \tag{8}$$

with $Y_\varnothing$ the empty-team outcome. Marginal interventional effects need not sum to the total when events interact: if two events are jointly necessary, each receives the full effect they produce together, so rescaling credits to close the gap would destroy exactly the information that the gap carries. A large gap flags a trajectory whose outcome depends on interactions rather than on individually pivotal events.

3.4 CALIBRATED GENERATIVE ORACLE

A single LLM judge is biased and high-variance on multi-agent processes (Venkataramani et al., 2026). CGO averages L self-consistent samples (Wang et al., 2023) from a committee of judges from different model families, $\hat{Y} = \frac{1}{|\mathcal{M}|}\sum_{J\in\mathcal{M}}\frac{1}{L}\sum_{l=1}^{L} J^{(l)}(\tau)$ with $\mathcal{M} = \{\text{GLM-5.2}, \text{GPT-4o}\}$ (Hurst et al., 2024), which dilutes, though it does not remove, the self-preference of the behavior model's own family. A monotone map is fit by isotonic regression on a verifiable anchor set $\mathcal{A}$ of size n,

$$g = \arg\min_{g\in\mathcal{G}_\uparrow} \sum_{i\in\mathcal{A}} \big(g(\hat{Y}_i) - V_i\big)^2, \tag{9}$$

and a query is abstained when the committee dispersion $\sigma(\tau)$ exceeds the conformal threshold

$$q_{1-\alpha} = \text{the } \lceil (n+1)(1-\alpha)\rceil\text{-th smallest of } \{\sigma_i\}_{i\in\mathcal{A}}, \qquad \text{abstain} \iff \sigma(\tau) > q_{1-\alpha}. \tag{10}$$

For a query exchangeable with the anchors, split conformal prediction gives $\Pr[\text{abstain}] \le \alpha$ (Angelopoulos & Bates, 2023). The guarantee is on the abstention rate; it does not by itself bound error on retained queries, which instead rests on dispersion being predictive of error, an empirical property we measure in Table 5.

Proposition 4 (Bias of calibrated credit)**.** *Suppose the calibrated oracle is ϵ-conditionally unbiased on the distributions of factual and counterfactual continuations, $|\mathbb{E}[g(\hat{Y}) - V \mid \textit{continuation law}]| \le \epsilon$ for both. Then the plug-in estimate of $\Delta(e_t)$ built from non-abstained continuations has bias at most 2ϵ relative to the verifier-defined credit.*

The proof substitutes $g(\hat{Y})$ for Y in both terms of equation 2 and applies the triangle inequality (Appendix A.1). The assumption is the substantive part: calibration on $\mathcal{A}$ controls bias on the *anchor* distribution, and transferring it to counterfactual continuations of open-ended tasks is an assumption of calibration transfer, which we cannot verify on OpenQA because it has no verifier. We state it explicitly, and it is what lets EVENTLEDGER score tasks where exact message-level methods such as C3 and CCPO (Chen et al., 2026; Li et al., 2026) have no readout at all.

3.5 REWARD COMPOSITION

The per-event reward combines calibrated credit with auxiliary terms over the event and its graph neighborhood,

$$\begin{aligned} r(e_t) = \lambda_\Delta \Delta(e_t) + \underbrace{\big(\gamma\Phi(G_t) - \Phi(G_{t-1})\big)}_{\text{potential-based progress}} + \lambda_g F_{\text{ground}}(e_t) \\ - \lambda_r C_{\text{red}}(e_t) - \lambda_c C_{\text{con}}(e_t) - \lambda_\$ C_{\text{cost}}(e_t) + \lambda_s S_{\text{stop}}(e_t), \end{aligned} \tag{11}$$

with potential $\Phi(G_t) = V_\theta^{\text{EVENTLEDGER-S}}(G_t)$ and $\Phi(G_T) = 0$. Because the progress term is potential-based, it leaves the set of optimal policies of the remaining reward unchanged (Ng et al., 1999); it speeds credit propagation without introducing a new optimum. The other terms are not potential-based and *do* change the objective, deliberately: F_{ground} rewards evidence-supported events, C_{red} and C_{con} penalize events that the same interventions identify as redundant or contradiction-inducing, and C_{cost} charges tokens, tool calls and latency (Appendix A.2). C_{red} charges only events that are both non-pivotal and duplicates of an earlier event, so a novel but unhelpful exploration step is not penalized as redundant. The stopping reward comes directly from the force-continue operator,

$$S_{\text{stop}}(e_t) = \mathbb{1}[\kappa(e_t)=\text{STOP}]\Big(\underbrace{\beta_\$ \widehat{\text{Cost}}_{\text{CONT}}}_{\text{saved cost}} - \underbrace{\big[\hat{Y}_{\text{CONT}} - \hat{Y}\big]_+}_{\text{forgone gain}}\Big), \tag{12}$$

so a stop earns reward when continuing would have cost money without improving the calibrated outcome, and is penalized when continuing would have improved it.

3.6 DISTILLATION AND DEPLOYMENT

The teacher is accurate but expensive, so we distill it into EVENTLEDGER-S, a graph-aware PRM initialized from an open-weight Qwen-3.5 checkpoint (Qwen Team, 2026) with a relational graph-attention head over typed events and their parents. It is trained with a Huber loss on event rewards

and a Bradley-Terry loss over sibling counterfactuals within each operator family, masking abstained targets,

$$\mathcal{L} = \sum_{e \notin \mathcal{E}_{\text{abs}}} \mathcal{H}_\delta\big(\hat{r}_\theta(e) - r(e)\big) \; - \; \beta \sum_{e \succ e'} \log \varsigma\big(\hat{r}_\theta(e) - \hat{r}_\theta(e')\big), \tag{13}$$

where ς is the logistic function. At deployment EVENTLEDGER-S makes no API calls and serves three uses: pruning candidate traces at the event level, an early-stopping controller driven by equation 12, and dense rewards for PPO (Schulman et al., 2017) on the Qwen-3.5 orchestrator. For N rollouts of the same task, we use a leave-one-out group baseline (Ahmadian et al., 2024; Shao et al., 2024),

$$\begin{aligned} \hat{A}_{i,t} &= \frac{R_{i,\geq t} - \frac{1}{N-1}\sum_{j \neq i} R_j}{\text{std}(R_1, \ldots, R_N)}, \\ \mathcal{J}(\phi) &= \mathbb{E}\Big[\textstyle\sum_t \min\big(\varrho_t \hat{A}_{i,t},\ \text{clip}(\varrho_t, 1 \pm \epsilon_{\text{clip}})\hat{A}_{i,t}\big)\Big] - \beta_{\text{KL}}\,\text{KL}(\pi_\phi \| \pi_{\text{ref}}). \end{aligned} \tag{14}$$

where $R_{i,\geq t} = \sum_{t' \geq t} \hat{r}_\theta(e_{i,t'})$ is the reward-to-go of rollout i, $R_j = R_{j,\geq 1}$, and $\varrho_t = \pi_\phi(e_{i,t} \mid s_{i,t}) / \pi_{\phi_{\text{old}}}(e_{i,t} \mid s_{i,t})$. The baseline is computed from the other rollouts only, so it does not depend on the actions of rollout i and leaves the policy gradient unbiased; the division by the group standard deviation is a per-group rescaling of the step size. The API models remain the engine for generation, intervention and judging; all training is confined to the student and the orchestrator.

4 EXPERIMENTS

Setup. We ask five questions: does typed credit improve end-task performance (RQ1); which components drive the gain (RQ2); what does it cost and does the stopping signal save compute (RQ3); is the oracle reliable (RQ4); and does the student transfer (RQ5). Three regimes span the axis from verifiable to open-ended tasks. `code_math` consists of HumanEval, MBPP and GSM8K (Chen et al., 2021; Austin et al., 2021; Cobbe et al., 2021), whose training partitions also form the anchor set $\mathcal{A}$, disjoint from the evaluation problems. `sql` is Spider (Yu et al., 2018), scored by execution accuracy against reference queries on the SQLite databases. `openqa` is an open-ended question-answering suite with no programmatic verifier, scored by the calibrated oracle. Worker agents are GLM-5.2 in every row, so rows differ only in the orchestrator and, for trained rows, in the reward used to train it. Baselines form two families (Appendix A.5): inference-time orchestration with a GLM-5.2 orchestrator (single agent with CoT, self-consistency, multi-agent debate, AutoGen-style roles (Wang et al., 2023; Du et al., 2023; Wu et al., 2023)), and a Qwen-3.5 orchestrator trained with PPO (Schulman et al., 2017) under the same action space and compute budget with one of the following rewards: outcome reward (Shao et al., 2024), uniform credit, COMA (Foerster et al., 2018), Shapley credit (Shapley, 1953), a RULER-style trajectory judge (Corbitt et al., 2025), MASPRM, COSAC, CoFi-PGMA or HCAPO. PRS uses $K = 8$ CRN pairs, with m set so that replay jobs are about 40% of those of the full estimator; CGO uses $\alpha = 0.1$. All scores are single runs without significance tests. The ablation of Section 4.2 uses a separate development split and the student-rewarded orchestrator, and each of its rows retrains the affected components. Further details are in Appendix A.3.

4.1 MAIN RESULTS

Table 2 supports three observations and one caution.

Reward training is not free. Inference-time orchestration improves only modestly over a single agent (from 80.55 to at most 81.80), and four of the nine trained baselines finish *below* the single agent on average. The losses concentrate on GSM8K, which the single agent already solves at 95.00: eight of the nine trained baselines fall to between 84.44 and 90.00. A coarse signal rewards whatever else happened in successful episodes, and on a task that one agent already solves, the orchestration it learns is mostly overhead that occasionally breaks a correct solution. Only COSAC (94.44) and EVENTLEDGER (95.44) stay within one point of the single agent, and both assign credit through sequential or event-level counterfactuals rather than a trajectory-level signal.

The margin is where the typed signal has something to say. Against the strongest baseline, HCAPO, EVENTLEDGER ties on HumanEval, Spider and OpenQA and gains $+5.32$ on MBPP and $+5.55$ on GSM8K; the entire $+2.18$ average margin comes from `code_math`. This is consistent with where typed operators differ from message-level ones: code and math traces are dominated by tool calls

Table 2: **Main results (RQ1).** All rows use GLM-5.2 worker agents; inference-time rows use a GLM-5.2 orchestrator, trained rows a Qwen-3.5 orchestrator trained by PPO whose only difference is the reward. Spider is execution accuracy; OpenQA is the calibrated CGO score scaled to $[0, 100]$. **Bold**: best per column; underlined : second best; ties share a mark.

	code_math			sql	openqa	
Method	HumanEval (pass@1)	MBPP (pass@1)	GSM8K (acc.)	Spider (exec. acc.)	OpenQA (CGO score)	Avg.
Inference-time orchestration (GLM-5.2 orchestrator, no reward training)						
Single agent (CoT)	90.63	89.10	95.00	81.00	47.00	80.55
Self-Consistency (Wang et al., 2023)	**97.56**	87.50	94.44	80.00	46.67	81.23
Multi-Agent Debate (Du et al., 2023)	93.29	90.03	**95.44**	81.00	48.00	81.55
AutoGen orchestration (Wu et al., 2023)	**97.56**	96.88	93.56	74.00	47.00	81.80
Trained orchestrator (Qwen-3.5, PPO), differing only in the credit or reward signal						
Outcome reward (Shao et al., 2024)	**97.56**	89.58	86.67	73.33	53.33	80.09
Uniform credit	**97.56**	93.75	90.00	73.33	40.00	78.93
COMA (Foerster et al., 2018)	95.83	89.58	85.56	**86.67**	53.33	82.19
Shapley credit (Shapley, 1953)	**97.56**	91.67	88.89	80.00	53.33	82.29
RULER (trajectory judge) (Corbitt et al., 2025)	**97.56**	91.67	87.78	**86.67**	46.67	82.07
MASPRM (Yazdani et al., 2025)	**97.56**	95.83	88.89	73.33	40.00	79.12
COSAC (Deshmukh et al., 2026)	**97.56**	97.92	94.44	80.00	46.67	83.32
CoFi-PGMA (Tong & Ben-Gal, 2026)	87.50	89.58	84.44	73.33	46.67	76.30
HCAPO (Tan et al., 2026)	**97.56**	93.75	89.89	**86.67**	**60.00**	85.57
EVENTLEDGER (teacher rewards)	**97.56**	**99.07**	**95.44**	**86.67**	**60.00**	**87.75**
EVENTLEDGER-S (student rewards)	**97.56**	**99.07**	**95.44**	**86.67**	**60.00**	**87.75**

(execution, tests) and critiques, whose counterfactuals, such as corrupting an argument or genericizing a critique, a message-deletion operator cannot express. On Spider and OpenQA we see no advantage over HCAPO.

The student matches the teacher end to end. Orchestrators trained with student and teacher rewards reach identical scores on every benchmark, although the student agrees with the per-event rewards of the teacher only at $\rho = 0.80$ (Table 6). The two statements are compatible: group-normalized advantages are invariant to the scale and offset of the rewards, and the student is trained with a ranking loss, so reward errors that preserve the ordering within a group change the update little. We read the result as parity at this evaluation resolution, not as evidence that distillation is lossless.

Caution on resolution. Trained-orchestrator scores on Spider and OpenQA move in steps of 6.67 points (one problem in fifteen), so a single problem separates many rows in those columns. We therefore make no claim from differences or ties smaller than that step, and treat the code_math columns as the primary evidence (Appendix A.4).

4.2 ABLATIONS

Table 3 separates two groups of components. Removing any credit-estimation or policy-optimization component (typed operators, CRN, the leave-one-out baseline, the leverage ranking) costs points on all three regimes: -1.33 on code_math, -5.67 to -6.67 on sql and -8.67 to -13.67 on openqa. The reward and oracle components, in contrast, are neutral on the two verifiable regimes and matter only on openqa, which is what their design predicts: calibration and abstention act only when the readout is a judge, and shaping matters most where the terminal signal is weakest. The largest single drops are on openqa (-13.34 without shaping and -13.67 without the leave-one-out baseline or the leverage ranking), consistent with a noisy judged readout being where variance control and a dense progress signal have the most to do. The ranking loss of the student matters only where margins are thin (-1.67 on sql, -3.34 on openqa).

Two patterns call for caution. Several pairs of ablations are identical in every column (typed operators and CRN; the leave-one-out baseline and the leverage ranking; calibration and abstention): at our development-split resolution they change the same problems, and we do not claim to separate their contributions. And removing the stopping reward leaves accuracy unchanged, because it is a cost term; its effect on cost is examined in Section 4.3.

Table 3: **Ablation (RQ2)** on the development split, one component removed at a time, regime-averaged scores. The absolute numbers are not comparable with Table 2 (different split). The stopping reward changes cost rather than accuracy (Section 4.3).

Variant	code_math	sql	openqa	Avg.
EVENTLEDGER (full)	**93.33**	**86.67**	**46.67**	**75.56**
Credit estimation and policy optimization (Sections 3.2, 3.3 and 3.6)				
w/o typed operators (message-only TCO)	92.00	80.00	38.00	70.00
w/o CRN pairing (Eq. equation 5)	92.00	80.00	38.00	70.00
w/o leave-one-out baseline (Eq. equation 14)	92.00	81.00	33.00	68.67
w/o leverage ranking (random m events)	92.00	81.00	33.00	68.67
Reward and oracle (Sections 3.4 and 3.5)				
w/o potential shaping (Eq. equation 11)	**93.33**	**86.67**	33.33	71.11
w/o stopping reward (Eq. equation 12)	**93.33**	**86.67**	**46.67**	**75.56**
w/o oracle calibration (raw judge)	**93.33**	**86.67**	40.00	73.33
w/o conformal abstention	**93.33**	**86.67**	40.00	73.33
Distillation (Section 3.6)				
w/o ranking loss (Eq. equation 13)	**93.33**	85.00	43.33	73.89

Table 4: **Cost (RQ3).** *Top*: deployment cost per task of each trained orchestrator (tokens in thousands); token reduction is relative to the untrained GLM-5.2 orchestrator. *Bottom*: cost of computing rewards over the training corpus (3,621 events). The per-task figures of the full counterfactual estimator are API-replay-equivalent estimates. Full table in Appendix A.6.

Deployment						
Orchestrator	API calls↓	Tokens (K)↓	Tool calls↓	Latency (s)↓	Token red. (%)↑	Avg. score↑
Untrained (GLM-5.2)	4.09	12.634	1.05	17.65	0.00	–
Outcome reward	**2.00**	**5.553**	1.05	**7.51**	**56.05**	80.09
RULER	3.95	10.736	**0.98**	16.00	15.02	82.07
MASPRM	3.80	10.400	1.00	15.50	17.68	79.12
EVENTLEDGER (teacher rewards)	3.18	9.771	1.03	13.33	22.66	**87.75**
EVENTLEDGER-S (+ stop controller)	**2.00**	6.038	1.00	**7.51**	52.21	**87.75**
Reward computation						
Scorer	Reward queries over the corpus			Per-task API cost of scoring		
Full counterfactual (every event)	6,608 replay jobs			20.59 calls, 234.3K tokens, 363 s		
EVENTLEDGER teacher (top-m)	2,628 replay jobs (39.8%)			budgeted, $O(mK)$		
EVENTLEDGER-S	3,621 local event scores			**0 API calls**		

4.3 COST AND EARLY STOPPING

Table 4 separates two costs that are easy to conflate. At *deployment*, the cheapest orchestrator is the outcome-reward one (56.1% fewer tokens than untrained), and it is cheap for a bad reason: with one terminal signal, the easiest way to raise expected reward is to do less, paid for with the lowest GSM8K score among the untrained and outcome-trained rows. EVENTLEDGER-S with its stop controller nearly matches that cost (52.2% reduction, identical API calls and latency) at the best average score, while the teacher-trained orchestrator without the controller saves 22.7%. Because the student-trained and teacher-trained orchestrators reach identical scores, we attribute most of this difference to the stop controller: the stopping signal is invisible in the accuracy ablation but visible in cost. For *reward computation*, intervening on every event would cost 234.3K tokens per task, about 18× the deployment cost of the untrained orchestrator, whereas the leverage budget replays 39.8% of the jobs and the student scores every event locally.

4.4 ORACLE RELIABILITY AND DISTILLATION

Each stage of the oracle does a different job (Table 5). The committee improves ranking and accuracy over a single judge. Isotonic calibration halves ECE ($0.32 \rightarrow 0.15$) without improving ranking; correlation dips slightly ($0.67 \rightarrow 0.65$) because the monotone step function merges nearby scores into ties, which is expected of calibration. Conformal abstention alone raises accuracy on what remains ($76.67 \rightarrow 83.33$) by declining 6.67% of queries, within the $\alpha = 10\%$ budget. That gain is

Table 5: **Oracle reliability (RQ4)** on a held-out verifiable anchor split ($\alpha = 0.1$). Retention is the fraction of non-abstained queries; Corr. and Acc. are on retained queries.

Oracle	Corr. ↑	Acc. ↑	ECE ↓	Retain (%)
Single judge (GLM-5.2)	0.62	73.33	0.44	100.00
Committee	0.67	76.67	0.32	100.00
+ isotonic	0.65	76.67	0.15	100.00
+ conformal (CGO)	**0.70**	**83.33**	**0.12**	93.33

Table 6: **Distillation and transfer (RQ5).** Agreement (ρ) and sibling rank accuracy against the teacher; RL gain in average-score points; cost relative to teacher scoring.

EventLedger-S on	ρ ↑	Rank ↑	Gain ↑	Cost
In-domain	**0.80**	**86.67**	**1.80**	0.05×
GPT-5.5 traces	0.75	83.33	1.60	0.05×
Claude orchestrator	0.71	83.00	1.50	0.05×
`code_math→openqa`	0.66	75.67	1.20	0.05×

Table 7: **MBPP task 16: harmful early stop.** Stopping at e_4 saves cost but changes a verified success into failure.

	Continue	**Stop at e_4**
Verifier score	**1.00**	**0.00**
Events / replay cost	9 / \$0.04492	4 / \$0.00995
Cost saved / correctness lost	–	\$0.03497 / 1.00000

$$S_{\text{stop}}(e_4) = \underbrace{0.03497}_{\text{cost saved}} - \underbrace{1.00000}_{\text{correctness lost}} = -0.96503$$

EventLedger decision: CONTINUE

the empirical evidence that committee dispersion predicts error, a property the conformal guarantee does not itself supply.

Distillation degrades along the axis one would expect (Table 6): agreement falls from 0.80 in-domain to 0.75 on traces from another generator and 0.71 for another orchestrator backbone, and it falls most *across regimes*, to 0.66 when a student trained on `code_math` scores `openqa`. The typed structure transfers across models more easily than across task families, so what the student learns is partly a property of the event distribution of the task.

Case study. Table 7 applies the stopping reward of equation 12 to MBPP task 16. Stopping at e_4 would save 77.85% of the replay cost, but it removes the verifier-backed solution, so the forgone gain dominates the saved cost and the controller continues.

5 Conclusion and Outlook

We introduced EventLedger, which replaces a broadcast outcome reward with interventional credit on typed orchestration events. Typed operators turn every event, including the decision to stop, into a scored counterfactual question; paired and budgeted rollouts keep the estimate unbiased on replayed events at 39.8% of the full replay cost; a calibrated committee with conformal abstention extends scoring to tasks without a verifier; and a distilled student delivers the signal at deployment without API calls. Empirically, the finer signal helps most where coarse rewards teach a policy orchestration that the task does not need, and the stopping reward, although invisible in accuracy, is decisive for cost. These results have clear limits: all numbers are single runs, the Spider and OpenQA splits move in steps of 6.67 points, Proposition 4 rests on calibration transferring from verifiable anchors to open-ended continuations, which cannot be checked without a verifier, and events outside the top-m set receive student-predicted rather than replayed credit (Appendix A.7).

Several directions follow. Group interventions on high-gap event sets could turn the conservation audit into interaction-aware credit; learned or composed operators would let credit ask richer counterfactual questions; a student trained on a mixture of regimes is a natural candidate for a general-purpose orchestration reward, since transfer is weakest across task families; and budget-conditioned stopping and cost terms would let one orchestrator serve different price points, turning typed credit into a direct handle on the cost of multi-agent inference.

AI USE STATEMENT

AI models are used in two ways in this work. First, API-based AI model calls are part of the proposed methodology: they are used for worker-agent generation, counterfactual replay, intervention evaluation, and judge-based scoring within the EVENTLEDGER pipeline, as described in section 3 and section 4. Second, AI tools were used for manuscript polishing, including improving wording, grammar, clarity, and readability. All final research claims, method design, theoretical statements, experiments, analyses, and reported results were reviewed and approved by the authors.

ETHICS STATEMENT

This work studies reward assignment and orchestration for multi-agent LLM systems. The experiments use public benchmark tasks in coding, math, text-to-SQL, and open-ended question answering, and do not involve human subjects, personally identifiable information, private user data, or clinical decision-making. The proposed method relies on API-based language models and judge models for generation, intervention, and evaluation; such models may inherit biases, produce incorrect judgments, or vary across providers and deployment settings. We therefore treat calibrated judge scores and event-level rewards as training and analysis signals rather than ground-truth measures of human preference or factual correctness. EventLedger is intended to improve the auditability and cost-awareness of multi-agent orchestration, but downstream deployment should include task-specific validation, monitoring for failure modes, and safeguards against over-reliance on automated judgments.

REPRODUCIBILITY STATEMENT

We submit the code as part of the supplementary materials. We describe the orchestration event graph, typed counterfactual operators, perturb-rollout estimator, calibrated generative oracle, reward composition, distillation objective, PPO training objective, baselines, evaluation protocol, ablations, and cost measurements in the main text and appendix. The paper specifies the benchmark suites, evaluation metrics, model roles, replay budget, committee construction, conformal abstention setting, and student-training losses used in our experiments. We also include implementation details, baseline definitions, cost accounting, proof sketches, and limitations in Appendix A.3, Appendix A.5, Appendix A.6, Appendix A.1, and Appendix A.7.

REFERENCES

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting reinforce-style optimization for learning from human feedback in llms. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12248–12267, 2024.

Anastasios N Angelopoulos and Stephen Bates. Conformal prediction: A gentle introduction. *Foundations and Trends in Machine Learning*, 16(4):494–591, 2023.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Mert Cemri, Melissa Z Pan, Shuyi Yang, Lakshya A Agrawal, Bhavya Chopra, Rishabh Tiwari, Kurt Keutzer, Aditya Parameswaran, Dan Klein, Kannan Ramchandran, et al. Why do multi-agent LLM systems fail? In *Advances in Neural Information Processing Systems*, volume 38, 2026.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Yanjun Chen, Yirong Sun, Hanlin Wang, Jinghan Wang, Xinming Zhang, Xiaoyu Shen, Wenjie Li, and Wei Zhang. Exact is easier: Credit assignment for cooperative LLM agents. *arXiv preprint arXiv:2603.06859*, 2026.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Kyle Corbitt, Saumya Gandhi, Angky William, Andie Jones, Brad Hilton, David Corbitt, and Bohdan Kovalevski. RULER: Relative universal LLM-elicited rewards, 2025.

Shripad Deshmukh, Jayakumar Subramanian, Raghavendra Addanki, and Nikos Vlassis. COSAC: Counterfactual credit assignment in sequential cooperative teams. *arXiv preprint arXiv:2604.17693*, 2026.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. *arXiv preprint arXiv:2305.14325*, 2023.

Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Zhitao He, Zijun Liu, Peng Li, Yi R Fung, Ming Yan, Ji Zhang, Fei Huang, and Yang Liu. Advancing language multi-agent learning with credit re-assignment for interactive environment generalization. *arXiv preprint arXiv:2502.14496*, 2025.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. GPT-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

Douglas Lai. GLM-5.2: Zhipu AI's 1M-token open-weight coding and agentic model. Eigent Blog, 2026. URL `https://www.eigent.ai/blog/glm-5-2`.

Zhongyi Li, Wan Tian, Yikun Ban, Jinju Chen, Huiming Zhang, Yang Liu, and Fuzhen Zhuang. Counterfactual credit policy optimization for multi-agent collaboration. *arXiv preprint arXiv:2603.21563*, 2026.

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*, volume 2024, pp. 39578–39601, 2024.

Andrew Y Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *International Conference on Machine Learning*, pp. 278–287, 1999.

Qwen Team. Qwen3.5-Omni technical report. *arXiv preprint arXiv:2604.15804*, 2026.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Lloyd S Shapley. A value for n-person games. In H. W. Kuhn and A. W. Tucker (eds.), *Contributions to the Theory of Games II*, pp. 307–317. Princeton University Press Princeton, 1953.

Hui-Ze Tan, Xiao-Wen Yang, Hao Chen, Jie-Jing Shao, Yi Wen, Yuteng Shen, Weihong Luo, Xiku Du, Lan-Zhe Guo, and Yu-Feng Li. Hindsight credit assignment for long-horizon LLM agents. *arXiv preprint arXiv:2603.08754*, 2026.

Stela Tong and Elai Ben-Gal. CoFi-PGMA: Counterfactual policy gradients under filtered feedback for multi-agent LLMs. *arXiv preprint arXiv:2604.22785*, 2026.

King Yeung Tsang, Zihao Zhao, Vishal Venkataramani, Haizhou Shi, Zixuan Ke, Semih Yavuz, Shafiq Joty, and Hao Wang. Reward modeling for multi-agent orchestration. *arXiv preprint arXiv:2606.13598*, 2026.

Vishal Venkataramani, Haizhou Shi, Zixuan Ke, Austin Xu, Xiaoxiao He, Yingbo Zhou, Semih Yavuz, Hao Wang, and Shafiq Joty. MAS-ProVe: Understanding the process verification of multi-agent systems. *arXiv preprint arXiv:2602.03053*, 2026.

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, 2024.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations*, 2023.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, et al. AutoGen: Enabling next-gen LLM applications via multi-agent conversation. *arXiv preprint arXiv:2308.08155*, 2023.

Zhiheng Xi, Chenyang Liao, Guanyu Li, Zhihao Zhang, Wenxiang Chen, Binghai Wang, Senjie Jin, Yuhao Zhou, Jian Guan, Wei Wu, et al. AgentPRM: Process reward models for LLM agents via step-wise promise and progress. In *Proceedings of the ACM Web Conference 2026*, pp. 4184–4195, 2026.

Peiran Xu, Zhuohao Li, Xiaoying Xing, Guannan Zhang, Debiao Li, and Kunyu Shi. Hybrid reward normalization for process-supervised non-verifiable agentic tasks. *arXiv preprint arXiv:2509.25598*, 2025.

Milad Yazdani, Mahdi Mostajabdaveh, Zirui Zhou, and Ying Xiong. MASPRM: Multi-agent system process reward model. *arXiv preprint arXiv:2510.24803*, 2025.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, et al. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 3911–3921, 2018.

Chenchen Zhang. From reasoning to agentic: Credit assignment in reinforcement learning for large language models. *arXiv preprint arXiv:2604.09459*, 2026a.

Chenchen Zhang. Reinforcement learning for LLM-based multi-agent systems through orchestration traces. *arXiv preprint arXiv:2605.02801*, 2026b.

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*, 2025.

Mingchen Zhuge, Wenyi Wang, Louis Kirsch, Francesco Faccio, Dmitrii Khizbullin, and Jürgen Schmidhuber. Language agents as optimizable graphs. In *International Conference on Machine Learning*, 2024.

Jiaru Zou, Ling Yang, Jingwen Gu, Jiahao Qiu, Ke Shen, Jingrui He, and Mengdi Wang. ReasonFlux-PRM: Trajectory-aware PRMs for long chain-of-thought reasoning in LLMs. In *Advances in Neural Information Processing Systems*, volume 38, pp. 26764–26802, 2025.

A Appendix

A.1 Proofs

Proposition 3. Under Assumption 1, for each k the factual readout $\rho(G_T(s_t, e_t, u^{(k)}))$ is distributed as the continuation law of $\mathrm{do}(e_t \leftarrow e_t)$ and the counterfactual readout as that of $\mathrm{do}(e_t \leftarrow \tilde{e}_t^{(k)})$ with $\tilde{e}_t^{(k)} \sim \mathcal{I}_\kappa(e_t)$. By linearity, $\mathbb{E}[\widehat{\Delta}]$ equals the difference of the two expectations in equation 2, whatever the joint distribution of the pair. The K pairs are i.i.d., so $\mathrm{Var}(\widehat{\Delta}) = \frac{1}{K}\mathrm{Var}(\rho_{\mathrm{fac}} - \rho_{\mathrm{cf}}) = \frac{1}{K}(\mathrm{Var}\rho_{\mathrm{fac}} + \mathrm{Var}\rho_{\mathrm{cf}} - 2\mathrm{Cov})$; independent sampling has $\mathrm{Cov} = 0$, hence equation 6. □

Proposition 4. Write $\mu_{\mathrm{fac}}, \mu_{\mathrm{cf}}$ for the verifier-defined expectations and $\hat{\mu}_{\mathrm{fac}}, \hat{\mu}_{\mathrm{cf}}$ for those of $g(\hat{Y})$ on the non-abstained continuations. By assumption $|\hat{\mu}_{\mathrm{fac}} - \mu_{\mathrm{fac}}| \le \epsilon$ and $|\hat{\mu}_{\mathrm{cf}} - \mu_{\mathrm{cf}}| \le \epsilon$, so $|(\hat{\mu}_{\mathrm{fac}} - \hat{\mu}_{\mathrm{cf}}) - (\mu_{\mathrm{fac}} - \mu_{\mathrm{cf}})| \le 2\epsilon$. □ The bound is tight when the two biases have opposite signs, and it is 0 when the bias of the oracle is the same on factual and counterfactual continuations, which is the favourable case because credit is a difference. The assumption is conditional on the continuation law; marginal calibration on $\mathcal{A}$ alone does not imply it.

A.2 Reward terms

$F_{\mathrm{ground}}(e_t)$ is the fraction of claims in e_t entailed by an earlier observation. $C_{\mathrm{red}}(e_t) = \mathbb{1}[|\widehat{\Delta}(e_t)| < \eta] \cdot \mathbb{1}[\exists j < t : \; e_j \equiv e_t]$ charges events whose removal does not matter *and* that duplicate an earlier event, so an unhelpful but novel exploration step is not penalized as redundant. $C_{\mathrm{con}}(e_t)$ counts downstream events that must be revised because of e_t, identified by the drop operator. C_{cost} is the provider's cost of the tokens, tool calls and latency of e_t. The progress term uses the value estimate of the student with $\Phi(G_T) = 0$, the condition under which policy invariance holds for episodic tasks.

A.3 Implementation

PRS uses $K = 8$ CRN pairs; the shared seed and sampling parameters are passed to the API for both members of a pair, and pairs whose factual replay does not reproduce the recorded continuation are kept but treated as independently sampled. The leverage budget is set so that replay jobs are $2{,}628$ of the $6{,}608$ of the full estimator (39.8%). CGO uses committee $\{\text{GLM-5.2}, \text{GPT-4o}\}$, $\alpha = 0.1$, and an anchor set drawn from the training partitions of HumanEval, MBPP and GSM8K; Table 5 is computed on a held-out anchor partition. EventLedger-S is a Qwen-3.5 checkpoint with a relational graph-attention head; the orchestrator is optimized by PPO with the leave-one-out group baseline of equation 14.

A.4 Evaluation protocol and resolution

All rows of Table 2 share the GLM-5.2 worker agents, the action space and the per-task compute budget. The ablation (Table 3) uses a development split and the student-rewarded orchestrator, and each row retrains the affected components. Scores are single runs; we do not report significance. Because trained-orchestrator scores on Spider and OpenQA move in steps of 6.67 points, we treat differences below one step in those columns as ties. In Table 6, the RL gain is the average-score improvement in the target setting obtained by PPO with EventLedger-S rewards over the orchestrator before training.

A.5 Baselines

Inference-time orchestration, with a GLM-5.2 orchestrator: a single agent with chain-of-thought; self-consistency over sampled solutions (Wang et al., 2023); multi-agent debate (Du et al., 2023); and an AutoGen-style role orchestrator (Wu et al., 2023). *Trained orchestrators*, with Qwen-3.5 and PPO (Schulman et al., 2017), differing only in reward: outcome reward (Shao et al., 2024); uniform credit, broadcasting the outcome to every event; the counterfactual baseline of COMA (Foerster et al., 2018); Shapley attribution over events (Shapley, 1953); a RULER-style trajectory judge (Corbitt et al., 2025); MASPRM (Yazdani et al., 2025); COSAC (Deshmukh et al., 2026); CoFi-PGMA (Tong & Ben-Gal, 2026); and HCAPO (Tan et al., 2026). All trained baselines use the same GLM-5.2 workers and compute budget.

A.6 Full cost table

Two readings of this table are worth making explicit. The savings of the outcome-reward orchestrator do not come from a stopping signal, since it has none, but from the policy learning to do less; its GSM8K score is the price. And the student is the only method that scores every event of the corpus, precisely because scoring is local and free; its reward-query count is therefore a coverage figure, not a cost.

Table 8: **All cost measurements.** "Reward queries" counts event scorings or replay jobs over the 3,621-event training corpus, depending on the method. Token reduction is relative to the untrained orchestrator. The full counterfactual row is an API-replay-equivalent estimate of *scoring* cost, not a deployment cost, and is therefore not comparable with the deployment columns of the other rows.

Method	API calls	Tokens (K)	Tool calls	Latency (s)	Reward queries	Token red. (%)
Untrained orchestrator	4.09	12.634	1.05	17.65	–	0.00
Outcome reward	2.00	5.553	1.05	7.51	978	56.05
RULER	3.95	10.736	0.98	16.00	1,919	15.02
MASPRM	3.80	10.400	1.00	15.50	3,500	17.68
Full counterfactual (scoring, est.)	20.59	234.285	21.55	363.44	6,608	−1,754.40
EVENTLEDGER (teacher rewards; top-m scoring)	3.18	9.771	1.03	13.33	2,628	22.66
EVENTLEDGER-S (+ stop controller)	2.00	6.038	1.00	7.51	3,621	52.21

A.7 SCOPE AND LIMITATIONS

Evaluation resolution. The Spider and OpenQA splits are small, so differences below one problem (6.67 points) in those columns are not rankings; our main evidence is `code_math`. All reported numbers are single runs, and seed variance and significance tests are the most important additions for a revised version.

Calibration transfer. Proposition 4 relies on calibration transferring from the anchor distribution to counterfactual continuations of open-ended tasks, which cannot be checked where there is no verifier.

Replay and budget. API nondeterminism weakens the CRN coupling, though not unbiasedness. Events outside the top-m set receive the credit predicted by the student rather than a replayed estimate.

Oracle. The judge committee includes the family of the behavior model and has two members, so its dispersion is a coarse uncertainty estimate; a larger committee would give a finer conformal score at higher cost.

Operators and baselines. The empty-team baseline $Y_{\varnothing}$ in equation 8 is 0 on every verifiable task we use, which makes the conservation gap equal to the credit sum minus the outcome; on tasks where a trivial answer can succeed it must be measured. The typed operator set is minimal rather than complete, and an event type without an operator falls back to deletion.