PREDICTIVE COORDINATION OF LLM AGENTS VIA SEMANTIC FAILURE ATTRIBUTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Teams of LLM agents often perform worse than a single agent, and recent benchmarks trace the losses to broken expectations and unkept commitments rather than to weak individuals. A coordinator that wants to avoid such failures must learn whom to trust. We prove it cannot learn this from outcomes: in a sequential-stage model of collaboration, a weak agent with a compatible partner and a strong agent with an incompatible partner induce identical success rates for every plan, so any estimator of individual competence suffers an error bounded away from zero, for any number of episodes and under any adaptive policy. Semantic attribution breaks the tie. If failures are labeled by cause through any invertible confusion channel M, the parameters become identifiable and estimation is $\sqrt{n}$-consistent at a price $\sigma_{\min}(M)^{-1}$, so the language model need only be *informative*, not accurate. On this result we build LPC (LLM-native Predictive Coordination), which gives an LLM coordinator belief filtering over competence and pairwise compatibility, predictive plan scoring, posterior-sampling exploration and a computable re-coordination threshold as tools under typed contracts. We then measure M for four LLMs on 96 multi-agent coding failures whose cause is fixed by construction. All are informative, but all share one bias: they identify a faulty patch almost perfectly and miss the interaction failure, blaming the implementer for what is in fact a broken agreement, which is precisely the failure outcome data also cannot see. Driven by these measured channels, LPC closes 65% to 70% of the reputation-to-oracle gap, beating plan-level Thompson sampling by a margin that grows with population size.

1 INTRODUCTION

Multi-agent LLM systems increasingly write software, analyze data and run long-horizon workflows (Wu et al., 2023; Hong et al., 2024; Qian et al., 2024), yet careful evaluations show that collaboration frequently *hurts*. On CooperBench, pairs of coding agents resolve on average 30% fewer tasks than a single agent doing both jobs alone, with failures dominated by broken expectations, unkept commitments and unproductive communication (Khatua et al., 2026). Agents exchange the right information yet fail to integrate it (Zhang et al., 2026), and multi-agent effects range from large gains to severe losses depending on task structure and single-agent capability (Kim et al., 2026). The question is therefore not whether multi-agent systems work, but *when to coordinate, with whom, under what plan, and when to change course*.

We view coordination as an inference-and-control problem. Whether B should verify A's patch depends on latent properties, such as how often A silently changes interfaces or whether A and B have kept contracts before, revealed only gradually through interaction traces. A coordinator must therefore *remember*, *attribute* (whose failure was it?), *predict*, *explore* and *react*. Existing work covers fragments of this loop: learned orchestration (Nielsen et al., 2026; Zhang et al., 2025b), topology optimization (Zhang et al., 2025a; Shen et al., 2025), partner profiles (Abdurahman et al., 2026; Zhou et al., 2026), and imagined futures for single agents (Hamidi et al., 2026). None closes the loop.

Trust is not identifiable from outcomes. Our starting point is a negative result about the first step. Every method that scores agents by how often their episodes succeed is estimating a quantity the data do not determine. A task succeeds when implementation, verification *and* integration all succeed, so a

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.

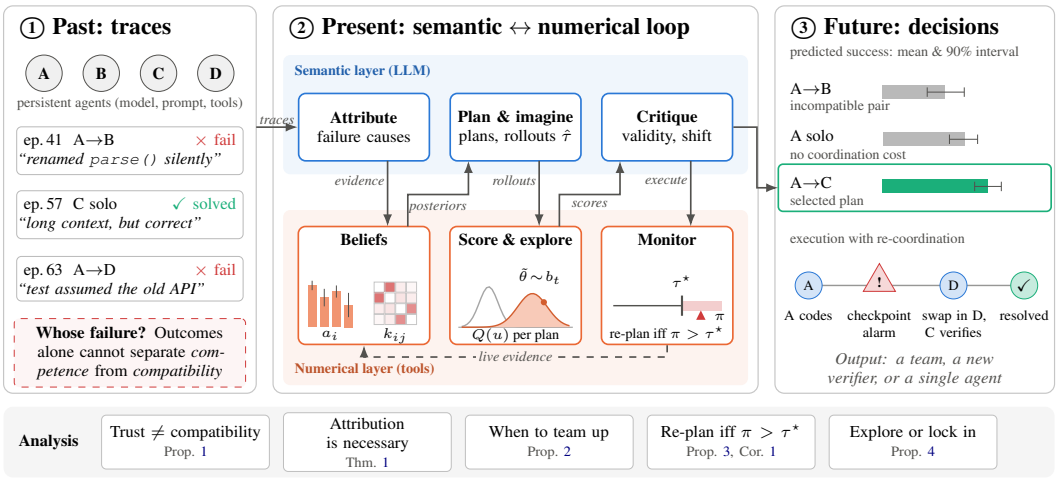


Figure 1: **LLM-native predictive coordination.** ① Trace outcomes cannot separate competence from compatibility. ② An LLM layer attributes causes, proposes and imagines plans and critiques decisions; a numerical layer maintains beliefs over competence a_i and compatibility k_{ij}, scores plans with posterior samples and monitors execution. ③ The coordinator chooses among teams, verifiers and solo work, and re-plans when $\pi > \tau^\star$. Bottom strip: the formal results. Values are illustrative.

success rate is a product, and a weak implementer with a compatible partner is indistinguishable from a strong implementer with an incompatible one, for every plan and any amount of data (Theorem 1(i)). This is a property of the observation channel, not of the estimator, so no exploration schedule and no richer model class removes it. What removes it is an observation reporting *which* stage failed, and such evidence lives in the trace: a diff, a message, a traceback. This is why the LLM belongs *inside* the estimation loop rather than around it, and Theorem 1(iii) prices its imperfection at a factor $\sigma_{\min}(M)^{-1}$, and nothing more.

A manager LLM doing all of this in prose fails, because LLMs do not accumulate evidence consistently over hundreds of episodes and their verbal confidence is not calibrated probability; a numerical controller treating agents as black boxes fails oppositely, by discarding the trace's semantics. Our design principle is a *typed division of labor*: LLMs interpret, propose, imagine and critique; numerical tools estimate, score, explore and stop; explicit contracts keep each honest (Figure 1).

Contributions. **(1)** A formulation of coordinating persistent LLM agents as belief-state control over plans, with textual traces as observations (Section 2). **(2)** LPC, which gives an LLM coordinator Bayesian filtering, predictive plan scoring, Thompson exploration and optimal re-coordination as tools under typed contracts (Section 3). **(3)** A separation theorem: trust cannot be learned from outcomes, but is learnable through any invertible attribution channel at a price $\sigma_{\min}(M)^{-1}$, with companion results on compatibility, team size and re-coordination (Section 4). **(4)** A measurement of M for four LLMs, on coding failures whose cause is fixed by construction rather than by human judgment: all are informative, and all are worst at exactly the failure outcome data also cannot see (Section 5). **(5)** Controlled experiments, driven by those measured channels, in which every prediction holds and LPC's gains grow with population size and survive misspecification (Section 6).

Related work. Multi-agent LLM frameworks (Wu et al., 2023; Hong et al., 2024; Qian et al., 2024; Du et al., 2024) and recent failure analyses (Zhu et al., 2025; Cemri et al., 2025; Zhang et al., 2026; Khatua et al., 2026; Kim et al., 2026; Huang et al., 2026) motivate our problem; orchestration and topology methods optimize the structure of collaboration for a task (Zhang et al., 2025b; Nielsen et al., 2026; Qian et al., 2025; Zhang et al., 2025a; Shen et al., 2025); partner models learn reputations, traits or theory of mind from history (Jøsang & Ismail, 2002; Gmytrasiewicz & Doshi, 2005; Abdurahman et al., 2026; Mu et al., 2026; Zhou et al., 2026); and imagined rollouts support planning for single agents (Hamidi et al., 2026). LPC instead conditions plans on uncertain beliefs about specific agents and pairings, proves that learning such beliefs requires semantic attribution, and closes the loop

through belief-triggered re-coordination (Kaelbling et al., 1998; Feldbaum, 1960). Appendix B extends this discussion.

2 Problem Formulation

We formalize the coordination of persistent AI agents as sequential decision making under partial observability. Our running example is collaborative software engineering, the setting of CooperBench (Khatua et al., 2026): four coding agents, A to D, repeatedly resolve issues in a shared repository, and for each issue a coordinator decides who works on it and how (Figure 5, Appendix C).

Agents and plans. A population $\mathcal{A} = \{1, \dots, N\}$ of LLM agents faces a stream of tasks q_t, each with an observable context x_t such as its programming language, module or dependency structure. Each agent is a pinned configuration (base model, system prompt, tool set and memory policy) whose identity persists across episodes, so evidence about an agent today is informative tomorrow.

Definition 1 (Coordination plan)**.** A *coordination plan* is a tuple $u = (S, \rho, G, \sigma, v)$ consisting of a team $S \subseteq \mathcal{A}$; an assignment ρ of roles or subtasks; a directed communication graph G over S whose edges carry interface contracts; a set of checkpoints σ at which commitments are tested; and a verification and fallback policy v specifying who checks whose work and who can take over.

In the running example (Figure 5b), A implements a token-rotation fix while B writes and runs the regression tests; a contract fixes the signature of `rotate_token()`, an interface test fires after implementation, and D stands by as fallback. Solo plans ($|S| = 1$) are always admissible, so *whether* to collaborate is part of the decision, and the plan space is combinatorial: implementer-verifier pairs alone number N^2 per context.

Traces: why outcomes are not enough. Executing u_t on q_t produces a *trace* τ_t: a time-ordered record of messages, commitments, tool calls, code artifacts and checkpoint results, with the outcome $Y_t \in \{0, 1\}$ and the incurred costs. Traces are the coordinator's only window onto its agents and are overwhelmingly textual. The outcome is one bit, but the *reason* for it lives in the trace: when B's regression test fails, $Y = 0$ is equally consistent with a wrong patch, a wrong test, or A silently changing the function signature after B tested against the agreed contract. Only the diff and the message log separate these, and the third says nothing bad about A or B individually. We write $\mathcal{H}_t = \{\tau_1, \dots, \tau_{t-1}\}$ for the persistent history, augmented during execution with the live partial trace.

Latent coordination state.

Definition 2 (Coordination state)**.** The *coordination state* z_t contains a competence $\theta^x_{i,r} \in [0, 1]$ for every agent i, role r and context x, and a compatibility $k^x_{ij} \in [0, 1]$ for every pair and context. The compatibility is the probability that the pair's contributions integrate correctly when each is individually correct. The state is constant except at unobserved change points, such as a model upgrade or a prompt edit.

The definition separates two objects reputation systems conflate: competence belongs to an agent, compatibility to a *pairing* (Proposition 1). Both are latent, and the coordinator needs a forward-looking prediction of how a plan will unfold, not a backward-looking success rate.

Objective. The coordinator maintains a belief $b_t(z) = \Pr(z \mid \mathcal{H}_t)$ and selects plans through a policy $u_t = \pi(b_t, x_t)$, which may revise the plan at checkpoints. With per-episode utility $U_t = R \cdot Y_t - C_t$, where C_t aggregates token, tool, latency and re-coordination costs, the goal is

$$\max_\pi \ \mathbb{E}\big[\textstyle\sum_{t=1}^{T} U_t\big]. \tag{1}$$

Under standard conditional-independence assumptions the belief is a sufficient statistic for (1) (Kaelbling et al., 1998; Nayyar et al., 2013). The problem is therefore a POMDP whose actions are plans and whose observations are traces (Figure 5a).

Why textbook solutions do not apply. The problem is intractable even for small teams, since decentralized POMDPs are NEXP-complete (Bernstein et al., 2002), and three features of LLM agents break the assumptions of approximate POMDP methods, each motivating a component of LPC. **(G1) Observations are text:** the likelihood $p(\tau_t \mid z_t, u_t)$ is unknown, so LPC replaces the trace with a typed, noisy attributed cause whose noise is modeled explicitly (Sections 3.1 and 3.2). **(G2) Plans are combinatorial:** an LLM proposes a few semantically sensible candidates, which tools score numerically (Section 3.3). **(G3) The state drifts and actions are informative** (Feldbaum, 1960): LPC explores by posterior sampling, forgets at a controlled rate, and re-coordinates when live evidence contradicts the plan (Sections 3.3 and 3.4).

3 LPC: LLM-Native Predictive Coordination

LPC implements the belief-state loop of Section 2 by dividing labor between a coordinator LLM and numerical tools (Figure 1), assigning each function to the component that performs it reliably. Four *interface contracts* make the division enforceable. **(C1)** The LLM may state only probabilities a tool returned. **(C2)** Tools consume typed records, never raw text. **(C3)** Every evidence record cites the trace events supporting it, so every belief update is auditable. **(C4)** Every tool output reports the contexts and amount of evidence behind it, so extrapolation is visible. C1 and C2 rule out the two failure modes of Section 1: confident prose masquerading as probability, and numerical updates blind to causal structure. Algorithm 1 and Table 2 in Appendix C give the full procedure; we describe its stages in order.

3.1 Semantic evidence extraction

After each episode, LLM-Extract reads the trace and emits a typed evidence record (Figure 6 in Appendix C, ②) holding the plan, the outcome, a cause label $L \in \{\mathsf{I}, \mathsf{T}, \mathsf{X}\}$, a short rationale, the cited event identifiers and the task context. The label names the *first* failing stage: implementation (I), verification (T), or the *interaction* between agents (X), such as an unkept commitment or a silent interface change. The labels mirror the taxonomy of Khatua et al. (2026), where expectation and commitment failures belong to the interaction rather than to either agent. Attribution is not assumed correct: before deployment LPC estimates a confusion matrix $\hat{M}_{\ell c} = \Pr(L = \ell \mid C = c)$ on an audited set of failures, and Theorem 1 shows any invertible $\hat{M}$ suffices, at a price governed by $\sigma_{\min}(\hat{M})$. Section 5 measures $\hat{M}$ for four LLMs.

3.2 Attribution-weighted belief filtering

LPC maintains independent Beta posteriors over implementation competences a_i^x, verification competences b_j^x, compatibilities k_{ij}^x and single-agent overload s_i^x, instantiating Definition 2 for implementer-verifier plans under the model of Section 4. A success adds one success count to every parameter involved; a failure labeled L is split across causes by its posterior responsibility,

$$w_c \;\propto\; \underbrace{\Pr(C = c \mid \bar{a}, \bar{b}, \bar{k})}_{\text{prior cause probability}} \cdot \underbrace{\hat{M}_{Lc}}_{\text{label likelihood}}, \qquad c \in \{\mathsf{I}, \mathsf{T}, \mathsf{X}\}, \tag{2}$$

where $\Pr(C = \mathsf{I}) = 1 - \bar{a}$, $\Pr(C = \mathsf{T}) = \bar{a}(1 - \bar{b})$ and $\Pr(C = \mathsf{X}) = \bar{a}\bar{b}(1 - \bar{k})$ are evaluated at posterior means. Each cause c contributes failure mass w_c to its own stage and success mass w_c to every earlier stage, which must have succeeded: a receives failure w_{I} and success $w_{\mathsf{T}} + w_{\mathsf{X}}$, b failure w_{T} and success w_{X}, and k failure w_{X}. This is assumed-density filtering (Minka, 2001): the exact posterior after an ambiguous failure is a mixture, and the update projects it back onto a product of Betas by matching expected counts. To track change points, the pseudo-counts an update touches are first discounted toward the prior, $P \leftarrow \gamma P + (1 - \gamma)P_0$, capping effective memory at $1/(1 - \gamma)$ observations at $O(1)$ cost.

A worked update (Figure 6, Appendix C) shows why the split matters. A and B are individually trusted (posterior means 0.82, 0.80) but little is known about the pair; without attribution a failure is blamed mostly on the individuals (0.41, 0.37), whereas the label X sends 0.68 to integration. After five such failures an outcome-only coordinator avoids A and B on every task, while LPC avoids only pairing them. This is the mechanism behind Theorem 1.

3.3 PREDICTIVE PLAN EVALUATION

LLM-PROPOSE receives the task and a belief summary (posterior means, credible intervals and evidence counts by role and context, plus flagged incompatible pairs), and returns K candidate plans differing in team, roles, contracts, checkpoints and fallback. Solo plans for the strongest agents are always added, so collaboration must earn its place (Proposition 2). The LLM addresses G2 by respecting constraints that are awkward to encode numerically, such as not splitting a database migration between two agents.

For each candidate u, LPC draws J parameter samples $\theta^{(m)} \sim b_t$ and asks LLM-ROLLOUT for a trajectory $\hat{\tau}^{(m)}$ consistent with each sample: a short account of how the plan would unfold, with its predicted outcome and cost. Conditioning rollouts on posterior samples keeps imagination tied to evidence: a pessimistic sample of k_{AB} yields a rollout in which A and B fail to integrate, and spells out the mechanism for the critic. A scoring tool maps rollouts to utilities $U = R \cdot \mathbf{1}\{\text{success}\} - \text{cost}$ and returns

$$Q(u) = \tfrac{1}{J} \textstyle\sum_{m=1}^{J} U\big(\hat{\tau}^{(m)}\big) - \lambda \, \mathrm{CVaR}_{\beta}\big(- U(\hat{\tau}^{(1:J)})\big). \tag{3}$$

Because each $\theta^{(m)}$ is a posterior sample rather than the posterior mean, choosing $\arg\max_u Q(u)$ with $J = 1$ is exactly Thompson sampling over plans (Thompson, 1933; Russo et al., 2018). This implements the value-of-information term of dual control without a hand-tuned bonus and avoids the lock-in of greedy reputation (Proposition 4); larger J and $\lambda > 0$ trade exploration for risk aversion.

3.4 CRITIQUE, MONITORING AND RE-COORDINATION

The top-scoring plan returns to the coordinator LLM with its predicted success, interval, evidence count and the contexts behind that evidence (C4). The critic accepts it, revises it (the revision is re-scored), or requests more evidence; its job is to catch misspecification the numbers cannot reveal, such as a compatibility learned only on Python tasks applied to a first Rust task. By C1 it may reject a plan but not overwrite its probabilities.

Only the first stage of the accepted plan is committed. Workers emit machine-readable commitments (files, interfaces, tests), which the checkpoints in σ test against the live trace. On an alarm the monitor computes the posterior failure probability π from the checkpoint's detection and false-alarm rates, and the threshold $\tau^\star$ of Proposition 3 from beliefs about the implementer, the best replacement and the verifier, then re-coordinates iff $\pi > \tau^\star$; the abandoned implementer receives a failure update of weight π. Because $\tau^\star$ is recomputed at every alarm, the same alarm triggers re-coordination when a strong replacement is cheap and is ignored otherwise (Figure 2c).

Per episode, LPC makes $K \cdot J + 3$ LLM calls, and its numerical tools run in milliseconds. In the controlled experiments, rollouts are exact draws from the sampled model ($J = 1$, $\lambda = 0$).

4 ANALYSIS: WHY SEMANTICS AND PREDICTION ARE BOTH NECESSARY

We analyze a stylized instance of the LPC decision problem to isolate the structural facts behind its design. Figure 2 previews three of the results, and proofs are in Appendix D.

Sequential-stage coordination model (SSCM). A plan $u = (i, j)$ assigns an implementer i and a verifier j; $u = (i, i)$ is a solo plan. In context x, execution passes through three independent stages: implementation ($\mathrm{Bern}(a_i^x)$), verification ($\mathrm{Bern}(b_j^x)$), and a third stage, which is integration ($\mathrm{Bern}(k_{ij}^x)$) for a pair and single-agent overload ($\mathrm{Bern}(s_i^x)$) for a solo plan. The task succeeds iff all stages succeed, which yields reward R; each participating agent costs c. The failure cause $C \in \{\mathsf{I}, \mathsf{T}, \mathsf{X}\}$ is the first failing stage. We write $p(i, j) = a_i b_j k_{ij}$ and $p(i, i) = a_i b_i s_i$, and drop x when it is clear from context.

Proposition 1 (Compatibility gap)**.** *Let $\hat{u}$ pair the implementer with the highest a_i with the remaining verifier with the highest b_j. For every $\epsilon \in (0, 1)$ there is a three-agent instance with $p(\hat{u}) \leq \epsilon \cdot \max_u p(u)$.*

Ranking agents by individual reputation is therefore not even a constant-factor approximation, which formalizes the observation of Khatua et al. (2026) that capable agents can fail as teammates.

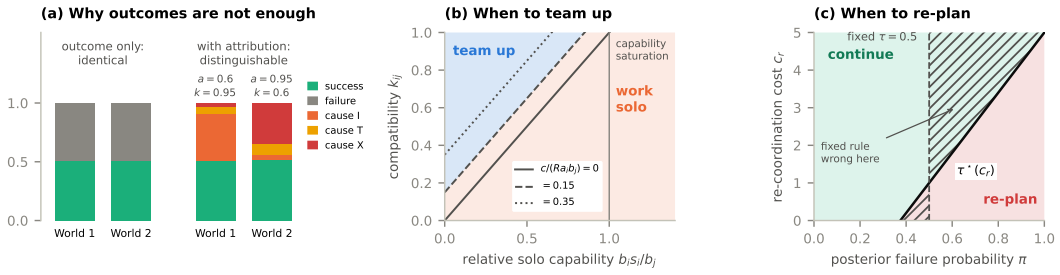


Figure 2: **Theory at a glance** (exact model computations). (a) Two worlds with identical success rates: a weak implementer with a compatible partner, and a strong one with an incompatible partner. Outcomes cannot tell them apart; failure causes can (Theorem 1). (b) Team versus solo regions (Proposition 2); cost lifts the boundary and beyond capability saturation ($b_i s_i / b_j \geq 1$) collaboration never pays. (c) Optimal re-coordination regions (Proposition 3; R=10, p'=0.6, q=0.8); a fixed threshold errs in the hatched region.

Theorem 1 (Separation: outcomes versus semantic attribution)**.** *Let $\theta = \{a_i, b_j, k_{ij}, s_i\}$ lie in the open unit cube, and let attribution labels be drawn from a confusion matrix M with $M_{\ell c} = \Pr(L = \ell \mid C = c)$.*

1. ***Outcome-only data cannot learn trust.*** *A continuum of parameters $\theta' \neq \theta$ induces the same outcome distribution for every plan. Hence there is $\Delta_\theta > 0$ such that every estimator $\hat{a}_i$ built from any number n of outcome-only episodes, under any adaptive plan selection, satisfies $\max_{\vartheta \in \{\theta, \theta'\}} \mathbb{E}_\vartheta |\hat{a}_i - a_i(\vartheta)| \geq \Delta_\theta$.*

2. ***Attribution restores identifiability.*** *If M is invertible, the law of (Y, L) under a plan determines its parameters. The symmetric channel M_α, which reports the true cause with probability α, is invertible iff $\alpha \neq 1/3$.*

3. ***Price of noise.*** *The moment estimator of the cause distribution from n episodes of plan u is $\sqrt{n}$-consistent. Its asymptotic covariance has operator norm at most $\sigma_{\min}(M)^{-2}(1 - p(u))$, where $\sigma_{\min}(M_\alpha) = \frac{3\alpha - 1}{2}$.*

Without attribution, estimation error is bounded away from zero for any amount of data (Figure 2a); with attribution it decays as $O\big(1/(\sigma_{\min}(M)\sqrt{n})\big)$, so the LLM need only be informative, not accurate. Part (i) instantiates the familiar fact that a product of Bernoulli rates does not determine its factors; the mechanism is not exotic, but its *orbit* is exactly what coordination needs. The rescaling $a_i \mapsto \lambda a_i$, $k_{ij} \mapsto k_{ij}/\lambda$ trades competence against compatibility, so the unidentified direction is precisely "is this agent weak, or is this pairing bad?", the question deciding whether to retire an agent or merely stop pairing it. The claim is also delimited, since outcomes do identify $p(u)$ itself: attribution earns its cost only when the coordinator must generalize to an unseen pairing, the combinatorial regime of Definition 1.

Proposition 2 (Coordination-benefit threshold)**.** *The pair (i, j) has higher expected utility than the solo plan (i, i) iff $k_{ij} > \kappa^\star_{ij} := b_i s_i / b_j + c/(Ra_i b_j)$. Thus $\kappa^\star$ grows with the solo agent's own capability and with cost, and if $b_i s_i \geq b_j$, collaboration with j never pays.*

This capability-saturation effect is consistent with the evidence of Kim et al. (2026). It is why LPC always considers solo plans (Figure 2b).

Proposition 3 (Optimal belief-triggered re-coordination)**.** *Suppose a checkpoint alarms with probability ρ if implementation failed and $\varphi < \rho$ otherwise. Let $\pi = \frac{(1-a_i)\rho}{(1-a_i)\rho + a_i\varphi}$, let q be the success probability of the remaining stages, and let p' be the success probability after replacing the implementer at cost $c_r + c$. Re-coordination is optimal iff $\pi > \tau^\star := 1 - (Rp' - c_r - c)/(Rq)$. A fixed threshold τ loses exactly $Rq|\pi - \tau^\star|$ whenever π lies between τ and $\tau^\star$.*

Corollary 1 (Plug-in re-coordination)**.** *If $\tau^\star$ and π are computed from estimates $(\hat{p}', \hat{q}, \hat{\pi})$, the per-alarm loss is at most $R\big(|\hat{p}' - p'| + |\hat{q}(1 - \hat{\pi}) - q(1 - \pi)|\big)$.*

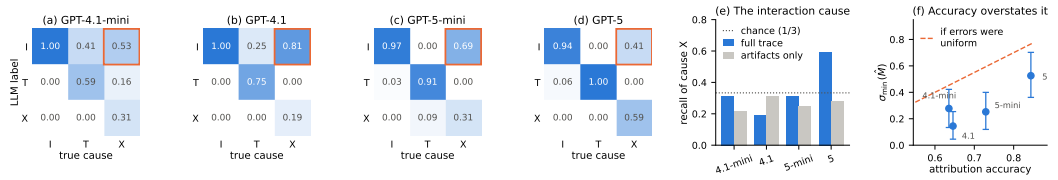


Figure 3: **What real LLMs get wrong when they attribute coordination failures** (n=96 failed episodes per model; ground-truth cause fixed by construction). (a) to (d) Measured channels $\hat{M}_{\ell c} = \Pr(L{=}\ell \mid C{=}c)$, columns summing to one. The boxed cell is every model's dominant error: an interaction failure X reported as an implementer failure I. (e) Recall of X with and without the inter-agent messages; deleting the record of the agreement costs three of the four models, most of all the strongest. (f) $\sigma_{\min}(\hat{M})$ against accuracy (95% cluster bootstrap over the 16 scenarios). Dashed: $\sigma_{\min}$ if errors were uniform. All four fall well below it, and the two orderings disagree.

The threshold couples beliefs about three agents with costs, so no fixed heuristic tracks it (Figure 2c). Its plug-in loss vanishes as posteriors concentrate, which by Theorem 1 requires attribution. Finally, greedy selection can lock out the best agent after one unlucky failure and incur linear regret (Proposition 4, Appendix D), which motivates Thompson sampling in LPC.

5 Measuring the Attribution Channel

Theorem 1 makes LPC depend on one empirical quantity: the attributor's confusion matrix M, through $\sigma_{\min}(M)$. If real language models attribute coordination failures at chance then $\sigma_{\min}(M) = 0$ and LPC inherits the impossibility of Theorem 1(i), so we measure M directly before evaluating any coordinator.

A corpus whose ground truth is not an opinion. Measuring M requires failures whose true cause is known. Human adjudication would make the ground truth itself an estimate, and its noise would sit exactly where ours does, since the cases a model finds ambiguous are the ones an annotator does too. We instead *construct* the cause: 16 small software tasks, each with a module, an issue and an interface contract that an implementer (A) and a verifier (B) agree before work starts, with each cause of Section 3.1 instantiated twice: I, A's patch has a genuine logic bug; T, A's patch is correct and B's test is wrong; X, both artifacts are correct on their own terms but A moved a name, signature, return type or exception away from the agreed contract while B tested against it. Every instance is executed, so diffs and tracebacks are real program output, and automatic checks enforce that the correct patch with the correct test passes and that each injected fault genuinely fails. This yields 96 failed episodes balanced across causes with no annotator noise in the ground truth; the cost is that the tasks are small, which we revisit in Section 7. Four LLMs then label each trace, given the taxonomy and nothing about the corpus or the prior over causes (Appendix F).

Attribution is informative, so LPC is not vacuous. Every model is well clear of the degenerate channel: accuracy runs from 0.64 to 0.84 against a chance level of $1/3$, and $\sigma_{\min}(\hat{M})$ from 0.14 $[0.05, 0.25]$ to 0.53 $[0.36, 0.70]$, all four intervals excluding zero (Figure 3f; Table 3 gives every entry). The premise of Theorem 1(ii) holds in practice, and the strongest model brackets the $\alpha = 0.8$ that Section 6 assumes.

The errors are structured, and structured against us. Every model identifies a faulty patch almost perfectly (recall of I is 0.94 to 1.00) and a faulty test well (0.59 to 1.00), but all are far weaker on the interaction cause: recall of X is 0.19 to 0.59, and 77% to 100% of the mass lost from X lands on I (Figure 3a to d). Asked why a collaboration failed, current LLMs blame the implementer for what is actually a broken agreement. This is the same failure mode twice over: X is the cause outcome data cannot see (Theorem 1), that reputation systems misattribute to individuals (Proposition 1), that dominates the taxonomy of Khatua et al. (2026), and now also the one the attributor is worst at. It instantiates our Ethics Statement's fairness concern: an agent often cast as implementer absorbs blame owed to the pairing.

Table 1: Controlled testbed (N=6, T=1500, drift at $T/2$, α=0.8; 50 seeds): utility per episode, 95% bootstrap CI, and paired difference Δ from LPC. Ablations: Table 5.

Method	Utility ↑	Δ vs. LPC	Success
Random plan	2.26 [2.15, 2.36]	-2.46 [-2.57, -2.35]	0.409
Reputation ranking	2.87 [2.60, 3.15]	-1.84 [-2.11, -1.58]	0.487
Individual trust + attribution	3.79 [3.56, 4.03]	-0.92 [-1.12, -0.72]	0.541
Plan-level Thompson sampling	3.78 [3.64, 3.92]	-0.93 [-1.00, -0.86]	0.562
LPC (full)	4.71 [4.58, 4.85]	n/a	0.685
Oracle (true parameters)	5.55 [5.43, 5.67]	+0.83 [+0.78, +0.89]	0.772

Accuracy overstates identifiability. Because the errors are directional rather than uniform, accuracy and $\sigma_{\min}$ come apart: GPT-4.1 is *more* accurate than GPT-4.1-mini (0.65 vs. 0.64) yet has exactly half its $\sigma_{\min}$ (0.14 vs. 0.28), because it sends its entire X error mass to I. The symmetric channel with the same $\sigma_{\min}$ makes the gap concrete: GPT-5's 0.84 accuracy is worth only $\alpha_{\text{eff}} = 0.68$ of uniform attribution (Figure 3f). Since Theorem 1(iii) makes $\sigma_{\min}$, not accuracy, the quantity controlling the estimation rate, an attributor should be selected on it, and the two orderings do differ.

The record of the agreement is what carries X. Deleting the two inter-agent messages should destroy information about X, since the agreement is what was broken, and it does: recall of X falls for three of the four models, most of all for the strongest, which drops from 0.59 to 0.28 (Figure 3e). GPT-4.1 is the exception, rising from 0.19 to 0.31, consistent with its being the model most willing to take A's routine hand-off at face value. The better a model attributes, the more of its skill rests on reading the coordination rather than the code, so model quality is best spent on LLM-EXTRACT, not the planner; Appendix G draws out what this implies for where effort should go.

6 EXPERIMENTS

We ask whether the predictions of Section 4 hold when a coordinator learns from noisy, drifting data; whether the semantic factorization beats strong generic learners; whether the advantage survives the attribution channels measured in Section 5; and whether LPC is robust when its own assumptions fail. A controlled testbed with known ground truth is what lets us compare against an oracle and vary one structural quantity at a time.

Testbed and protocol. $N = 6$ persistent agents face $T = 1500$ tasks from two contexts under the model of Section 4, with $R = 10$ and $c = 1$. Competences are drawn from $\mathcal{U}[0.55, 0.97]$, and 30% of (pair, context) cells are incompatible ($k \sim \mathcal{U}[0.25, 0.55]$). Checkpoints detect failed implementations with probability 0.7 at a 0.1 false-alarm rate. Halfway through the stream, the strongest agent's competence silently drops to 0.3. LLM-EXTRACT is replaced by an attribution channel ($\alpha = 0.8$ unless varied, or a measured $\hat{M}$) and rollouts by exact posterior samples, which isolates the decision layer (Appendix H). Baselines are random plans; reputation ranking (Jøsang & Ismail, 2002); individual trust with attribution but no pairwise terms; plan-level Thompson sampling, with one arm per (plan, context) and no decomposition (Russo et al., 2018); and an oracle. Ablations remove one component at a time. Hyperparameters were chosen on development seeds disjoint from the 50 test seeds, and we report paired differences with 95% bootstrap intervals.

Main results. LPC attains 4.71 utility per episode against 2.87 for reputation ranking and 5.55 for the oracle, closing 69% of the gap (Table 1). The strongest generic learner, plan-level Thompson sampling, reaches only 3.78; LPC beats it by $+0.93$ $[0.86, 1.00]$, and even LPC *without* re-coordination by $+0.72$ $[0.66, 0.79]$. The advantage therefore comes from the factorized, attributed belief state, which shares evidence about an agent across all its partners. In the ablations (Table 5), every component except the solo option contributes significantly: exploration ($+0.44$), attribution ($+0.21$), compatibility ($+0.21$), belief-triggered re-coordination ($+0.21$) and forgetting ($+0.13$); re-coordinating on every alarm recovers only half the re-coordination benefit, as Proposition 3 predicts.

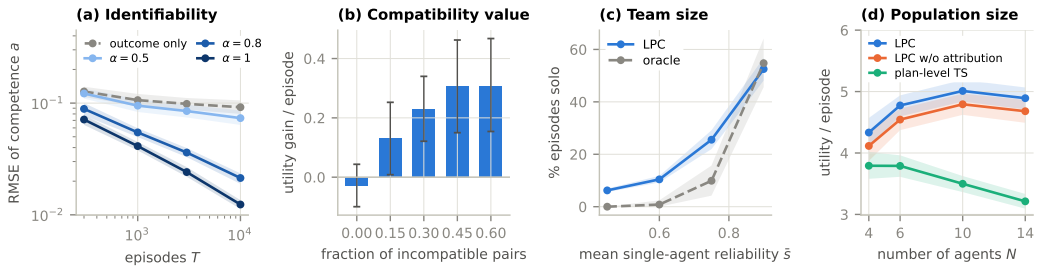


Figure 4: **The theory's predictions under learning** (95% bootstrap bands). (a) Outcome-only estimation error plateaus while attributed estimation keeps improving (Theorem 1). (b) The value of compatibility modeling grows with the prevalence of incompatible pairs (Proposition 1). (c) LPC shifts to solo work as single agents strengthen, tracking the oracle (Proposition 2). (d) Plan-level learning degrades as the population grows, while LPC improves.

The theory's predictions hold. From $T = 1000$ to $T = 10{,}000$ episodes, outcome-only estimation barely improves (RMSE $0.106 \to 0.092$), exactly as Theorem 1(i) predicts, while at $\alpha = 0.8$ the error falls from 0.055 to 0.021 (Figure 4a). Compatibility modeling is worthless when all pairs are compatible (-0.03 $[-0.10, 0.04]$) and worth $+0.31$ at 60% prevalence (Figure 4b), and as single agents strengthen LPC moves from teams to solo work much as the oracle does (Figure 4c). Plan-level learning must learn every plan separately, so its utility *falls* as the population grows, from 3.79 at $N = 4$ to 3.21 at $N = 14$, while LPC's advantage grows from $+0.54$ to $+1.68$ (Figure 4d). After the capability drop, reputation ranking keeps choosing the formerly best agent while LPC recovers within about 100 episodes (Figure 7a), and the belief-dependent $\tau^\star$, without tuning, matches the best *single* fixed threshold chosen in hindsight (Table 7).

The advantage survives real attribution channels. Replacing the idealized $\alpha = 0.8$ channel by each of the four measured in Section 5, emitting labels from $\hat{M}$ and giving the filter the same $\hat{M}$, leaves the conclusion intact (Table 4). LPC reaches 4.60 to 4.75, closing 65% to 70% of the reputation-to-oracle gap, beating plan-level Thompson sampling by $+0.82$ to $+0.97$. The paired gain from attribution falls from $+0.21$ under the idealized channel to between $+0.10$ $[0.01, 0.19]$ and $+0.25$ $[0.16, 0.34]$, and stays significant under all four; it rises with the labeller's accuracy, though neighbouring intervals overlap, so we read that as a trend rather than a ranking. LPC therefore does not require a frontier attributor: even the weakest channel we measured pays, since a coordinator that under-detects incompatibility still learns competence correctly, only converging on a bad pair more slowly.

Robustness to misspecification. If the filter assumes the wrong attribution accuracy, every $\hat{\alpha} \in [0.6, 1.0]$ still beats dropping attribution; only a severe underestimate ($\hat{\alpha} = 0.4$) does worse than ignoring the labels (Figure 7b). The environment can also violate the belief model, through asymmetric compatibility, unobserved difficulty shocks, or both; LPC remains the best non-oracle method, closing 57% to 69% of the gap and beating plan-level Thompson sampling by $+0.80$ to $+0.98$ (Table 6).

7 Discussion

Two gaps separate our evidence from a deployment claim. First, the attribution corpus is constructed: that removes annotator noise from the ground truth but keeps faults crisp, so $\sigma_{\min}(\hat{M})$ on repository-scale failures will fall below Table 3. Read those values as an upper bound on attribution quality available today, and expect the $\mathsf{X} \to \mathsf{I}$ bias to survive, since harder traces make the interaction cause harder. Second, the coordinator is evaluated in simulation: the measured channels now drive the belief update, but plan proposal and rollout remain surrogates and no claim here establishes an end-to-end resolve-rate gain (Appendix I pre-registers that study). Coordination should be a closed loop: language models read, propose and imagine; numerical tools remember, predict and decide; and the right output is sometimes a team, sometimes a different verifier, and sometimes one agent alone.

Reproducibility statement

Every number is reproducible from the supplementary material, and all assumptions and proofs are in Appendix D. The simulator, experiment driver and figure and table generators are in `sim/`, where each environment draws its parameters and all common random numbers from a single integer seed; `cd sim && python3 experiments.py && python3 make_figures.py` regenerates Section 6 exactly, on test seeds disjoint from the development seeds used to choose T and γ (Appendix H). The attribution study is in `sim/attribution/`, which ships the task library, the constructed traces and all 768 LLM labels, so Table 3 reproduces with no API call (Appendix F).

Ethics statement

This work involves no human subjects and no personal data: the controlled experiments run on synthetic parameters, and the attribution corpus is code we wrote. Two risks are worth naming. LPC keeps persistent quantitative beliefs about named agent configurations, which is a reputation system; interface contract C3, which makes every update cite the trace events behind it, is what keeps such judgments auditable rather than opaque, and the same machinery pointed at people rather than configurations would be workplace surveillance. Attribution error is also not uniform: Section 5 measures a bias that charges interaction failures to the implementer, and by Theorem 1 such a bias enters the estimates through M and does not wash out with more data, so $\hat{M}$ should be audited per configuration and per context rather than only in aggregate.

References

Suhaib Abdurahman, Etsuko Ishii, Katerina Margatina, Divya Bhargavi, Monica Sunkara, and Yi Zhang. Explicit trait inference for multi-agent coordination. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1670–1704. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-long.77.

Saaket Agashe, Yue Fan, Anthony Reyna, and Xin Eric Wang. LLM-Coordination: Evaluating and analyzing multi-agent coordination abilities in large language models. *arXiv preprint arXiv:2310.03903*, 2023.

Shipra Agrawal and Navin Goyal. Analysis of Thompson sampling for the multi-armed bandit problem. In *Conference on Learning Theory (COLT)*, 2012.

Tamer Başar and Geert Jan Olsder. *Dynamic Noncooperative Game Theory*. SIAM, 2nd edition, 1999.

Daniel S. Bernstein, Robert Givan, Neil Immerman, and Shlomo Zilberstein. The complexity of decentralized control of Markov decision processes. *Mathematics of Operations Research*, 27(4): 819–840, 2002.

Mert Cemri, Melissa Z. Pan, Shuyi Yang, Lakshya A. Agrawal, Bhavya Chopra, Rishabh Tiwari, Kurt Keutzer, Aditya Parameswaran, Dan Klein, Kannan Ramchandran, Matei Zaharia, Alexandros G. Dimakis, and Ion Stoica. Why do multi-agent LLM systems fail? *arXiv preprint arXiv:2503.13657*, 2025.

Matthew Chang et al. PARTNR: A benchmark for planning and reasoning in embodied multi-agent tasks. In *International Conference on Learning Representations (ICLR)*, 2025.

Wei Chen, Yajun Wang, and Yang Yuan. Combinatorial multi-armed bandit: General framework and applications. In *International Conference on Machine Learning (ICML)*, 2013.

Yijiang River Dong, Tiancheng Hu, Zheng Hui, Caiqi Zhang, Ivan Vulić, Andreea Bobu, and Nigel Collier. Value of information: A framework for human–agent communication. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 42879–42896. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-long.1987.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 11733–11763. PMLR, 2024.

A. A. Feldbaum. Dual control theory. I. *Avtomatika i Telemekhanika*, 21(9):1240–1249, 1960.

Piotr J. Gmytrasiewicz and Piotr Doshi. A framework for sequential planning in multi-agent settings. *Journal of Artificial Intelligence Research*, 24:49–79, 2005.

Shayan Mohajer Hamidi, Linfeng Ye, and Konstantinos N. Plataniotis. DreamPhase: Offline imagination and uncertainty-guided planning for large-language-model agents. In *International Conference on Learning Representations (ICLR)*, 2026.

Sirui Hong et al. MetaGPT: Meta programming for a multi-agent collaborative framework. In *International Conference on Learning Representations (ICLR)*, 2024.

Yue Huang, Yu Jiang, Wenjie Wang, Haomin Zhuang, Xiaonan Luo, Yuchen Ma, Zhangchen Xu, Zichen Chen, Nuno Moniz, Zinan Lin, Pin-Yu Chen, Nitesh V. Chawla, Nouha Dziri, Huan Sun, and Xiangliang Zhang. Emergent social intelligence risks in generative multi-agent systems. arXiv preprint arXiv:2603.27771, 2026. URL https://arxiv.org/abs/2603.27771.

Zhong-Ping Jiang, Andrew R. Teel, and Laurent Praly. Small-gain theorem for ISS systems and applications. *Mathematics of Control, Signals and Systems*, 7:95–120, 1994.

Audun Jøsang and Roslan Ismail. The beta reputation system. In *Proceedings of the 15th Bled Electronic Commerce Conference*, 2002.

Audun Jøsang, Roslan Ismail, and Colin Boyd. A survey of trust and reputation systems for online service provision. *Decision Support Systems*, 43(2):618–644, 2007.

Leslie Pack Kaelbling, Michael L. Littman, and Allan R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1–2):99–134, 1998.

Arpandeep Khatua, Hao Zhu, Peter Tran, Arya Prabhudesai, Frederic Sadrieh, Johann K. Lieberwirth, Xinkai Yu, Yicheng Fu, Michael J. Ryan, Jiaxin Pei, and Diyi Yang. CooperBench: Why coding agents cannot be your teammates yet. arXiv preprint arXiv:2601.13295, 2026. URL https://arxiv.org/abs/2601.13295.

Yubin Kim, Ken Gu, Chanwoo Park, Chunjong Park, Samuel Schmidgall, A. Ali Heydari, Yao Yan, Zhihan Zhang, Yuchen Zhuang, Yun Liu, Mark Malhotra, Paul Pu Liang, Hae Won Park, Yuzhe Yang, Xuhai Xu, Yilun Du, Shwetak Patel, Tim Althoff, Daniel McDuff, and Xin Liu. Capable language models can outgrow the benefits of collaboration. *Nature Machine Intelligence*, 8(7): 1157–1172, 2026. doi: 10.1038/s42256-026-01268-y.

George J. Mailath and Larry Samuelson. *Repeated Games and Reputations: Long-Run Relationships*. Oxford University Press, 2006.

David Q. Mayne, James B. Rawlings, Christopher V. Rao, and Pierre O. M. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, 2000.

Richard D. McKelvey and Thomas R. Palfrey. Quantal response equilibria for normal form games. *Games and Economic Behavior*, 10(1):6–38, 1995.

Thomas P. Minka. Expectation propagation for approximate Bayesian inference. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2001.

Dov Monderer and Lloyd S. Shapley. Potential games. *Games and Economic Behavior*, 14(1): 124–143, 1996.

Chunjiang Mu, Ya Zeng, Qiaosheng Zhang, Kun Shao, Chen Chu, Hao Guo, Danyang Jia, Zhen Wang, and Shuyue Hu. Adaptive theory of mind for LLM-based multi-agent coordination. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 29608–29616, 2026. doi: 10.1609/aaai.v40i35.40204.

Ashutosh Nayyar, Aditya Mahajan, and Demosthenis Teneketzis. Decentralized stochastic control with partial history sharing: A common information approach. *IEEE Transactions on Automatic Control*, 58(7):1644–1658, 2013.

Stefan Nielsen, Edoardo Cetin, Peter Schwendeman, Qi Sun, Jinglue Xu, and Yujin Tang. Learning to orchestrate agents in natural language with the conductor. In *International Conference on Learning Representations (ICLR)*, 2026.

H. Vincent Poor and Olympia Hadjiliadis. *Quickest Detection*. Cambridge University Press, 2008.

Chen Qian et al. ChatDev: Communicative agents for software development. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024.

Chen Qian et al. Scaling large language model-based multi-agent collaboration. In *International Conference on Learning Representations (ICLR)*, 2025.

Daniel J. Russo, Benjamin Van Roy, Abbas Kazerouni, Ian Osband, and Zheng Wen. A tutorial on Thompson sampling. *Foundations and Trends in Machine Learning*, 11(1):1–96, 2018.

Xu Shen, Yixin Liu, Yiwei Dai, Yili Wang, Rui Miao, Yue Tan, Shirui Pan, and Xin Wang. Understanding the information propagation effects of communication topologies in LLM-based multi-agent systems. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 12347–12361. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.emnlp-main.623.

William R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294, 1933.

Xingyao Wang et al. OpenHands: An open platform for AI software developers as generalist agents. In *International Conference on Learning Representations (ICLR)*, 2025.

Qingyun Wu, Gaurav Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awaan, and Chi Wang. AutoGen: Enabling next-gen LLM applications via multi-agent conversation. *arXiv preprint arXiv:2308.08155*, 2023.

John Yang, Carlos E. Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. SWE-agent: Agent-computer interfaces enable automated software engineering. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

John Yang et al. SWE-smith: Scaling data for software engineering agents. *arXiv preprint arXiv:2504.21798*, 2025.

Guibin Zhang et al. Cut the crap: An economical communication pipeline for LLM-based multi-agent systems. In *International Conference on Learning Representations (ICLR)*, 2025a.

Jiayi Zhang et al. AFlow: Automating agentic workflow generation. In *International Conference on Learning Representations (ICLR)*, 2025b.

Yuzhe Zhang, Feiran Liu, Yi Shan, Xinyi Huang, Xin Yang, Yueqi Zhu, Xuxin Cheng, Cao Liu, Ke Zeng, Terry Jingchen Zhang, and Wenyuan Jiang. SILO-BENCH: A scalable environment for evaluating distributed coordination in multi-agent LLM systems. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 29379–29398. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-long.1354.

Ruiwen Zhou, Maojia Song, Xiaobao Wu, Sitao Cheng, Xunjian Yin, Yuxi Xie, Zhuoqun Hao, Wenyue Hua, Liangming Pan, Soujanya Poria, and Min-Yen Kan. Epistemic context learning: Building trust the right way in LLM-based multi-agent systems. arXiv preprint arXiv:2601.21742, 2026. URL https://arxiv.org/abs/2601.21742.

Kunlun Zhu, Hongyi Du, Zhaochen Hong, Xiaocheng Yang, Shuyi Guo, Zhe Wang, Zhenhailong Wang, Cheng Qian, Xiangru Tang, Heng Ji, and Jiaxuan You. MultiAgentBench: Evaluating the collaboration and competition of LLM agents. *arXiv preprint arXiv:2503.01935*, 2025.

A THE USE OF LARGE LANGUAGE MODELS

LLMs enter this work in two distinct ways, and we disclose both.

As an object of study. LPC places an LLM inside its decision loop by design. Section 5 calls four models (GPT-4.1-mini, GPT-4.1, GPT-5-mini and GPT-5) as the LLM-EXTRACT component, producing the 768 labels that Table 3 is computed from; those labels are released with the supplementary material. The controlled experiments of Section 6 call no LLM at run time: the filter is driven either by an idealized channel or by a measured $\hat{M}$.

As a research and writing assistant. Addressing the tasks the ICLR policy asks authors to report, an LLM assistant was used to *generate synthetic data sets* (it implemented the 16-task corpus of Appendix F from fault specifications we set, and every instance is then validated by execution rather than by inspection); to *implement methods* (the simulator, the attribution pipeline, and the analysis and figure code); to *design and give feedback on experiments* (it proposed the trace-richness ablation and the $\sigma_{\min}$-versus-accuracy comparison, and we chose what to run); to *interpret results* (it drafted descriptions of the measured channels, which we checked against the raw outputs); and for writing, LaTeX and figure construction. From the same list, LLMs were *not* used to develop the theoretical model or the conceptual framework, to formulate the mathematical claims, to supply ingredients for or write any proof in Appendix D, to propose the paper's hypotheses, for translation, or for qualitative and thematic analysis.

Verification. We have reviewed all AI-assisted work. Concretely: every numerical claim in Sections 5 and 6 was re-derived from the released raw results and checked against the text; the attribution corpus enforces two invariants programmatically and the build fails if either breaks, which caught two injected faults that the verifier's tests did not actually exercise (Appendix F); the extension of the simulator to general confusion matrices was confirmed to leave the previously reported numbers bit-identical on the same seeds; and the trace corpus is byte-reproducible across runs. All research claims, theorem statements, proofs and experimental designs are the authors' own, and the authors take full responsibility for the content of the paper, including any part produced with LLM assistance.

B EXTENDED RELATED WORK

Multi-agent LLM systems and their failures. Conversation frameworks (Wu et al., 2023), role-structured software teams (Hong et al., 2024; Qian et al., 2024) and multi-agent debate (Du et al., 2024) established multi-agent LLM pipelines. Benchmarks now probe collaboration directly (Agashe et al., 2023; Zhu et al., 2025; Chang et al., 2025; Zhang et al., 2026; Khatua et al., 2026). Failure taxonomies (Cemri et al., 2025; Khatua et al., 2026) and studies of emergent collective risks (Huang et al., 2026) show that team-level failures cannot be reduced to individual ones, and a matched-compute study finds that the value of collaboration depends on task structure and single-agent capability (Kim et al., 2026). Proposition 2 gives a mechanism for this dependence, and LPC is a coordinator that acts on it.

Orchestration and topology. Several lines of work optimize the *structure* of collaboration:

- workflow search (Zhang et al., 2025b);
- learned natural-language orchestration (Nielsen et al., 2026);
- DAG-structured scaling to large agent populations (Qian et al., 2025);
- communication pruning and topology design (Zhang et al., 2025a; Shen et al., 2025).

These methods condition structure primarily on the task. LPC is complementary: it conditions structure on persistent, uncertain beliefs about *specific* agents, and revises it online as evidence arrives.

Partner modeling, trust and reputation. Bayesian reputation systems aggregate binary feedback (Jøsang & Ismail, 2002; Jøsang et al., 2007), and interactive POMDPs maintain beliefs over other agents' models (Gmytrasiewicz & Doshi, 2005). Recent LLM work infers partner traits (Abdurahman

et al., 2026) or theory-of-mind order (Mu et al., 2026) from interaction history, and builds trust-aware peer profiles (Zhou et al., 2026). Theorem 1 shows why outcome-based trust is fundamentally non-identifiable in teams, which explains why reading traces semantically is not optional. LPC further couples such beliefs to prediction and control.

Prediction, planning and decision theory. LPC draws on several classical foundations:

- belief-state planning (Kaelbling et al., 1998);
- the common-information approach to decentralized control (Nayyar et al., 2013);
- receding-horizon control (Mayne et al., 2000);
- quickest detection (Poor & Hadjiliadis, 2008);
- dual control and Thompson sampling (Feldbaum, 1960; Thompson, 1933; Russo et al., 2018; Agrawal & Goyal, 2012).

Exact decentralized POMDPs are intractable (Bernstein et al., 2002), so we use these theories to structure tractable tools rather than to solve the full problem. DreamPhase (Hamidi et al., 2026) uses imagined rollouts with uncertainty gating for a single LLM agent, and value-of-information analyses guide when agents should ask for clarification (Dong et al., 2026). We extend predictive planning to the *interaction* between agents. Combinatorial bandits (Chen et al., 2013) study selection over structured action sets; our factorized belief can be viewed as a semantically informed combinatorial bandit whose arms share parameters through attribution.

C Algorithm and Tool Interface

Table 2: Division of labor in LPC. The LLM never states a probability that no numerical tool returned; numerical tools never interpret raw text.

Function	Component	Output	Theory
Interpret the past	LLM	typed evidence, failure attribution	semantic reasoning
Maintain beliefs	numerical	posteriors: competence, compatibility	Bayesian filtering
Imagine futures	LLM + numerical	candidate plans, rollouts	generative world model
Evaluate futures	numerical	expected utility, risk, exploration	dual control
Adapt online	numerical + LLM	re-coordination decision	optimal stopping

Algorithm 1 LPC: LLM-native Predictive Coordination (one episode)

Require: task q_t with context x_t; population $\mathcal{A}$; history $\mathcal{H}_t$; belief b_t

$E_t \leftarrow$ LLM-Extract(Retrieve($\mathcal{H}_t, q_t$)) ▷ typed, cited evidence

$b_t \leftarrow$ Update-Beliefs($b_t, E_t; M, \gamma$) ▷ (2)

$\mathcal{U}_t \leftarrow$ LLM-Propose(q_t, Summarize(b_t)) $\cup$ {solo plans}

for $u \in \mathcal{U}_t$ **do**

  sample $\theta^{(1:J)} \sim b_t$; $\hat{\tau}^{(m)} \leftarrow$ LLM-Rollout($u, q_t, \theta^{(m)}$)

  $Q(u) \leftarrow$ Score($\hat{\tau}^{(1:J)}$)

$u^\star \leftarrow \arg\max_u Q(u)$; $u^\star \leftarrow$ LLM-Critique($u^\star, Q, b_t, x_t$)

execute first stage of $u^\star$; collect commitments and live trace

if checkpoint alarm **and** $\pi > \tau^\star(b_t)$ **then** ▷ Proposition 3

  re-coordinate: replace implementer by $\arg\max_{i'} \Pr(\text{success} \mid i', b_t)$

finish execution; $\mathcal{H}_{t+1} \leftarrow \mathcal{H}_t \cup \{\tau_t\}$; update b with attributed outcome

Tool interface. Tools return distributions and diagnostics, not bare scores. Each output carries an estimate, an uncertainty interval, the evidence basis (trace ids and counts), a validity region (the contexts it was estimated on), and the decision consequence of being wrong. The critic uses the validity region to detect distribution shift, and every tool call is logged, so each coordination decision can be audited against its evidence.

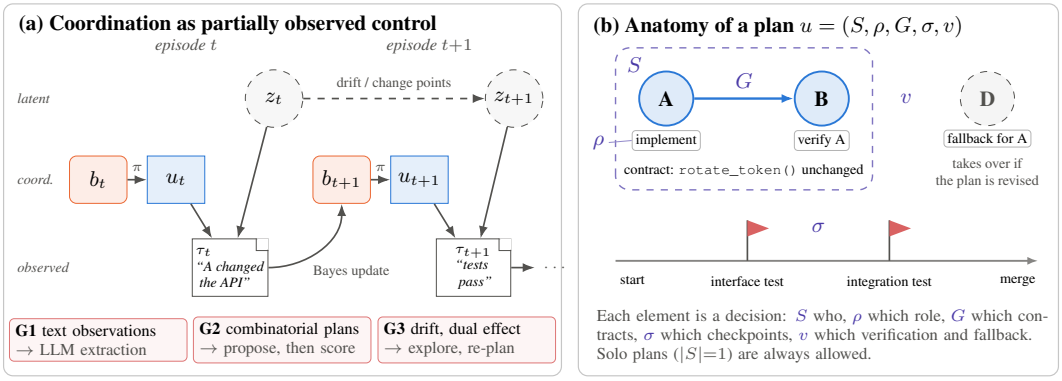


Figure 5: **Problem formulation.** (a) The coordinator acts on a belief b_t over the latent coordination state z_t (competences and pairwise compatibilities), chooses a plan u_t, and observes a mostly textual trace τ_t from which it updates its belief. The state drifts across episodes. Three gaps (G1 to G3) rule out textbook POMDP solutions and motivate the components of LPC. (b) The running example: A implements a token-rotation fix under an interface contract, B verifies, two checkpoints test commitments, and D is on standby.

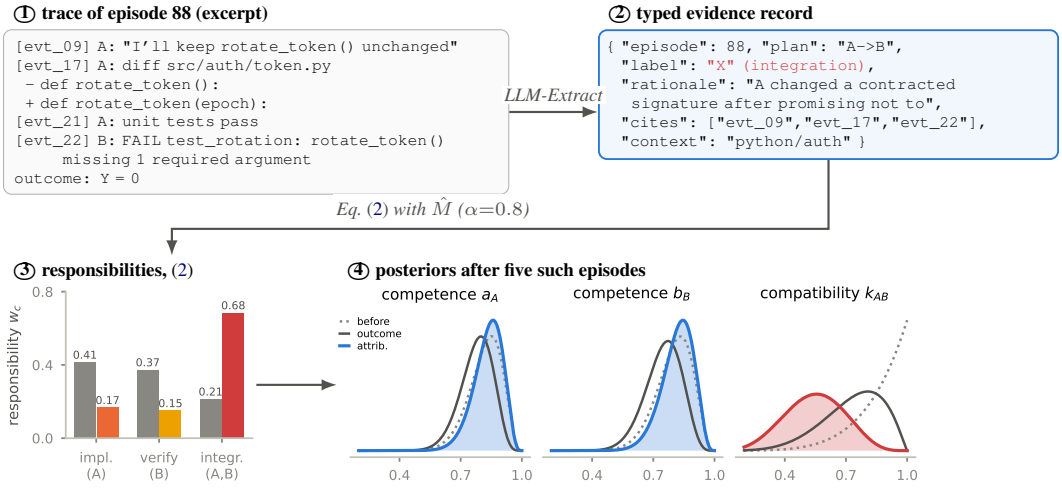


Figure 6: **Attribution-weighted belief filtering, worked example.** ① A failure caused by a silent interface change. All three causes are consistent with $Y{=}0$; only the diff and the message log reveal that the *interaction* failed and that neither A nor B is individually at fault. ② LLM-EXTRACT labels it an integration failure and cites the evidence. ③ Without attribution (gray), blame follows prior failure rates and falls mostly on the individual agents; with the label, it concentrates on integration. ④ After five such failures, outcome-only filtering downgrades both agents (A $0.82{\to}0.78$, B $0.80{\to}0.75$) and so begins to avoid A and B *everywhere*; attributed filtering leaves them intact and localizes the evidence on the pair (k_{AB}: $0.86{\to}0.55$), so LPC avoids only *pairing* them. All values are exact outputs of the LPC update.

D PROOFS

D.1 PROOF OF PROPOSITION 1

Take agents $\{1,2,3\}$ and a single context. Let $a = (1, \frac{1}{2}, \frac{1}{2})$, $b = (\frac{1}{2}, 1, \frac{1}{2})$, $k_{12} = k_{21} = \epsilon/2$, and all other compatibilities equal to 1; solo parameters $s_i \in [0,1]$ are arbitrary. Individual ranking selects implementer 1 (largest a) and, among the remaining agents, verifier 2 (largest b), so $p(\hat{u}) = 1 \cdot 1 \cdot \epsilon/2$. The plan $(1,3)$ achieves $p(1,3) = 1 \cdot \frac{1}{2} \cdot 1 = \frac{1}{2}$, hence $\max_u p(u) \geq \frac{1}{2}$ and $p(\hat{u}) = \epsilon/2 \leq \epsilon \max_u p(u)$. The expected-utility gap $R(\frac{1}{2} - \frac{\epsilon}{2})$ grows without bound in R. □

D.2 Proof of Theorem 1

(i) Fix an agent i and $\lambda > 0$, and define θ' by $a'_i = \lambda a_i$, $b'_i = \lambda b_i$, $k'_{ij} = k'_{ji} = k_{ij}/\lambda$ for all $j \neq i$, $s'_i = s_i/\lambda^2$, leaving every other parameter unchanged. Then for every $j \neq i$: $p'(i,j) = \lambda a_i\, b_j\, k_{ij}/\lambda = p(i,j)$, $p'(j,i) = a_j\, \lambda b_i\, k_{ij}/\lambda = p(j,i)$, and $p'(i,i) = \lambda^2 a_i b_i\, s_i/\lambda^2 = p(i,i)$; plans not involving i are unaffected. Since the outcome of a plan is $\mathrm{Bern}(p(u))$, all outcome laws coincide. θ' lies in the unit cube whenever $\max\{\max_j k_{ij}, \sqrt{s_i}\} < \lambda < \min\{1/a_i, 1/b_i\}$; for θ in the open cube this interval contains $\lambda = 1$ in its interior, giving a continuum of observationally equivalent parameters.

Lower bound. Fix one such θ' with $\lambda \neq 1$ and let $\Delta_\theta = |a_i - a'_i|/2 = |1-\lambda|a_i/2 > 0$. Under any (possibly adaptive) plan-selection rule, the joint law of the entire outcome sequence is built from per-plan outcome laws, which coincide under θ and θ'; hence the data have the same law P under both. For any estimator $\hat{a}_i$,

$$\mathbb{E}_\theta|\hat{a}_i - a_i| + \mathbb{E}_{\theta'}|\hat{a}_i - a'_i| = \mathbb{E}_P\big[|\hat{a}_i - a_i| + |\hat{a}_i - a'_i|\big] \geq |a_i - a'_i|,$$

so the larger of the two terms is at least Δ_θ, for every n.

(ii) Fix a pair plan $u = (i,j)$ and write (a,b,k) for its parameters. The true cause distribution on failure events is $q = (1-a,\ a(1-b),\ ab(1-k))^\top$, and $\Pr(Y=1) = abk = 1 - \mathbf{1}^\top q$. The observable vector $r_\ell = \Pr(Y=0, L=\ell)$ equals Mq. If M is invertible, $q = M^{-1}r$ and the parameters follow recursively: $a = 1-q_1$, $b = 1-q_2/a$, $k = 1-q_3/(ab)$, all well defined because $a, b \in (0,1)$. The solo case is identical with s in place of k. For the symmetric channel, $M_\alpha = \frac{3\alpha-1}{2}\, I + \frac{1-\alpha}{2}\, \mathbf{1}\mathbf{1}^\top$. Because $\mathbf{1}\mathbf{1}^\top$ has eigenvalue 3 on $\mathbf{1}$ and 0 on $\mathbf{1}^\perp$, M_α has eigenvalue 1 on $\mathbf{1}$ and $\frac{3\alpha-1}{2}$ (multiplicity two) on $\mathbf{1}^\perp$; it is invertible iff $\alpha \neq \frac{1}{3}$. When $\alpha = \frac{1}{3}$ the label is independent of the cause, and we are back in case (i).

(iii) Let $\hat{r}_n$ be the empirical frequencies of $\{Y=0, L=\ell\}$ over n i.i.d. episodes. By the multivariate CLT, $\sqrt{n}(\hat{r}_n - r) \Rightarrow \mathcal{N}(0, \Sigma_r)$ with $\Sigma_r = \mathrm{diag}(r) - rr^\top$, so $\hat{q}_n = M^{-1}\hat{r}_n$ satisfies $\sqrt{n}(\hat{q}_n - q) \Rightarrow \mathcal{N}(0, M^{-1}\Sigma_r M^{-\top})$. Since $0 \preceq \Sigma_r \preceq \mathrm{diag}(r)$ and the columns of M sum to one, $\|\Sigma_r\|_{\mathrm{op}} \leq \max_\ell r_\ell \leq \mathbf{1}^\top r = \mathbf{1}^\top q = 1 - p(u)$. Hence $\|M^{-1}\Sigma_r M^{-\top}\|_{\mathrm{op}} \leq \|M^{-1}\|^2_{\mathrm{op}}(1-p(u)) = \sigma_{\min}(M)^{-2}(1-p(u))$. M_α is symmetric with eigenvalues 1 and $\frac{3\alpha-1}{2} \leq 1$ for $\alpha \in (\frac{1}{3}, 1]$, so $\sigma_{\min}(M_\alpha) = \frac{3\alpha-1}{2}$, and perfect attribution ($M = I$) gives $1 - p(u)$. The parameters (a,b,k) are smooth functions of q on the open cube, so by the delta method they inherit $\sqrt{n}$-consistency, with covariance transformed by a fixed Jacobian. □

D.3 Proof of Proposition 2

$V(i,j) - V(i,i) = Ra_ib_jk_{ij} - 2c - (Ra_ib_is_i - c)$, which is positive iff $k_{ij} > (Ra_ib_is_i + c)/(Ra_ib_j) = \kappa^\star_{ij}$. Monotonicity in b_is_i and c is immediate. If $b_is_i \geq b_j$ then $\kappa^\star_{ij} \geq 1 + c/(Ra_ib_j) > 1 \geq k_{ij}$, so the pair is never preferred. □

D.4 Proof of Proposition 3

By Bayes' rule, $\Pr(\text{impl. failed} \mid \text{alarm}) = \pi$. Costs incurred before the alarm are common to both branches and cancel. Continuing succeeds iff the implementation has not failed and the remaining stages succeed, which are independent of the alarm given the implementation outcome; hence $V_c = R(1-\pi)q$. Replacing the implementer yields an independent fresh implementation, so $V_r = Rp' - c_r - c$. Then

$$V_r - V_c = Rp' - c_r - c - Rq + Rq\pi = Rq\,(\pi - \tau^\star),$$

which is positive iff $\pi > \tau^\star$. A fixed-threshold rule differs from the optimal action exactly when π lies between τ and $\tau^\star$, and the loss incurred is then $|V_r - V_c| = Rq|\pi - \tau^\star|$. For $\tau = 0$ the rule always re-coordinates, losing $Rq(\tau^\star - \pi)$ whenever $\pi < \tau^\star$. □

D.5 Proof of Corollary 1

From the proof of Proposition 3, $V_r - V_c = Rp' - c_r - c - Rq(1-\pi) = \Delta$. If $\hat{\Delta}$ and Δ have the same sign, the plug-in rule takes the optimal action and loses nothing. Otherwise it

loses $|\Delta|$, and because the signs differ, $|\Delta| \le |\Delta - \hat{\Delta}|$. Since the costs are known, $\hat{\Delta} - \Delta = R(\hat{p}' - p') - R\big(\hat{q}(1-\hat{\pi}) - q(1-\pi)\big)$, and the triangle inequality gives the bound. □

D.6 Greedy reputation lock-in

Proposition 4 (Greedy reputation lock-in)**.** *Consider two plans with success probabilities $p_1 > p_2 > 1/3$, uniform Beta priors, and a coordinator that always executes the plan with the highest posterior mean, breaking ties toward plan 1. With probability at least $(1-p_1)\,\eta > 0$, for a constant $\eta > 0$ that depends only on p_2, plan 1 is never executed after the first episode. The expected regret then grows linearly in the horizon.*

Proof. Let $\delta = p_2 - \frac{1}{3} > 0$. In episode 1 both posterior means equal $\frac{1}{2}$ and plan 1 is executed; it fails with probability $1 - p_1$, leaving its mean at $\frac{1}{3}$. Plan 1 is never executed again as long as plan 2's posterior mean stays strictly above $\frac{1}{3}$. After n executions of plan 2 with S_n successes, its mean is $(S_n + 1)/(n + 2) > \frac{1}{3}$ iff $S_n > (n-1)/3$.

Fix $m \in \mathbb{N}$ and consider the event that the first m executions of plan 2 all succeed (probability p_2^m), in which case $S_n = n > (n-1)/3$ for $n \le m$. For $n = m + l$, $S_n = m + B_l$ with $B_l \sim \mathrm{Bin}(l, p_2)$, and a violation requires $B_l \le (l - 2m - 1)/3 = l p_2 - t_l$ with $t_l = l\delta + \frac{2m+1}{3} > 0$. By Hoeffding's inequality, $\Pr(B_l \le l p_2 - t_l) \le \exp(-2t_l^2/l) \le \exp(-2l\delta^2)\exp(-\frac{4}{3}\delta(2m+1))$. A union bound over $l \ge 1$ gives

$$\Pr(\exists\, l \ge 1 : \text{violation} \mid \text{first } m \text{ succeed}) \le e^{-\frac{4}{3}\delta(2m+1)} \frac{e^{-2\delta^2}}{1 - e^{-2\delta^2}} =: B_m,$$

and $B_m \to 0$ as $m \to \infty$. Choosing m with $B_m \le \frac{1}{2}$, the event that plan 1 is never executed after episode 1 has probability at least $(1-p_1)\,p_2^m/2$, so $\eta = p_2^m/2$. On this event the per-episode regret is $R(p_1 - p_2)$, so expected regret is $\Omega(R(p_1 - p_2)(1-p_1)\eta\, T)$. □

E Extensions: Strategic and Dynamical Structure

The following results are not evaluated in this paper but indicate how the LPC belief state supports game-theoretic and stability reasoning when agents are heterogeneous or self-interested (Başar & Olsder, 1999; Mailath & Samuelson, 2006).

A plan is a mechanism. A plan u fixes participants, information flow, commitments and verification, and therefore induces a Bayesian game $\Gamma_t(u)$ among agents with types drawn from the coordinator's belief. The outer LPC problem $\max_u \mathbb{E}_{b_t}[W(\sigma^\star(\Gamma_t(u)))]$, with $\sigma^\star$ a response model (team-optimal, quantal response (McKelvey & Palfrey, 1995), or a policy learned from traces), is a receding-horizon mechanism-design problem.

Proposition 5 (Credible commitments)**.** *Suppose deviating from a declared commitment yields immediate gain ε, is detected with probability ϑ (the verification intensity chosen by the plan), and upon detection incurs penalty P and a loss ΔV of continuation value (e.g., reduced future allocation). Honoring the commitment is incentive compatible iff $\varepsilon \le \vartheta(P + \Delta V)$.*

Proof. The expected net gain from deviation is $\varepsilon - \vartheta(P + \Delta V)$. □

Persistent traces raise ΔV and LPC's verifier assignment raises ϑ; the semantic layer must still distinguish strategic deviation from incapacity before applying the penalty.

Proposition 6 (Aligned local objectives)**.** *Let $\Phi(x)$ be a team objective over joint actions and define difference utilities $U_i(x_i, x_{-i}) = \Phi(x_i, x_{-i}) - \Phi(\bar{x}_i, x_{-i})$ for fixed baselines $\bar{x}_i$. Then Φ is an exact potential (Monderer & Shapley, 1996): every unilateral improvement of U_i improves Φ by the same amount, so with finite action sets better-response dynamics terminate at a pure Nash equilibrium that is a local maximizer of Φ.*

Proof. $U_i(x_i', x_{-i}) - U_i(x_i, x_{-i}) = \Phi(x_i', x_{-i}) - \Phi(x_i, x_{-i})$ since the baseline term does not depend on x_i. □

Error propagation. Stack expectation, commitment and communication errors of all agents into e_t and consider the linearized interconnection $e_{t+1} = A(u)e_t + w_t$, where $A(u)$ encodes how the plan's topology propagates errors. If $g := \max_i \sum_j |A_{ij}(u)| < 1$, then $\|e_{t+1}\|_\infty \le g\|e_t\|_\infty + \|w_t\|_\infty$, hence $\limsup_t \|e_t\|_\infty \le \sup_t \|w_t\|_\infty/(1-g)$ for any initial error, a small-gain condition in the sense of Jiang et al. (1994). Verifier edges and narrower interfaces reduce row sums of $A(u)$ and thus serve as stabilizing design choices; estimating $A(u)$ from traces is left to future work.

F The Attribution-Channel Study

This appendix documents the measurement of Section 5. Code and data are in `sim/attribution/`: `scenarios.py` (the task library), `generate.py` (trace construction and validation), `label.py` (the LLM calls) and `analyze_attr.py` (the channel estimates and the testbed re-run).

Why the ground truth is constructed rather than adjudicated. $M_{\ell c} = \Pr(L = \ell \mid C = c)$ can only be estimated against a known C. The natural protocol, having humans adjudicate logged failures, makes the ground truth itself a noisy estimate, and its noise is concentrated exactly where ours is, because the cases that make I and X hard for a model are the cases that make them hard for an annotator. Adjudication noise biases $\hat{M}$ toward the identity whenever the annotator and the labeler share a failure mode, which would inflate $\sigma_{\min}(\hat{M})$ and flatter our conclusion. Constructing the fault removes this: we know C because we caused it. The price is external validity, which we state in Section 7.

Task library. Sixteen self-contained Python tasks, each a module of 6 to 26 non-blank lines once patched, an issue in the style of a bug report, and a one-line interface contract. The rendered traces run 52 to 120 lines (median 76). The domains are token rotation, retry backoff, CSV field parsing, LRU eviction, semantic-version comparison, half-open interval overlap, pagination bounds, token-bucket rate limiting, query-string merging, config layering, path normalization, batch chunking, integer money splitting, word-boundary truncation, topological sort and interval merging. Each carries a correct patch, a correct test, two buggy patches, two buggy tests and two contract-breaking patches, giving $16 \times 3 \times 2 = 96$ failed episodes balanced across the three causes.

Construction of each cause. I pairs a buggy patch with the correct test: the patch does not do what the issue asks (an off-by-one at an expiry boundary, an uncapped refill, a dropped final chunk) and a fair test catches it. T pairs the correct patch with a buggy test: the test asserts a convention the issue never established (remainder on the last share rather than the first, touching intervals reported as overlapping). X pairs a contract-breaking patch with the correct test: the patch is a reasonable implementation of the issue but departs from the agreed interface: a renamed entry point, a tuple where a list was agreed, reordered positional parameters, a keyword-only argument, a `ValueError` where a custom exception was agreed. In X neither artifact is wrong on its own terms, and the repair is to restore the agreement.

Validation. Every instance is materialized as a real package and run under pytest. Two conditions are enforced programmatically and the corpus is rejected if either fails: the correct patch with the correct test must pass, and every fault instantiation must fail. The second condition caught two cases during construction in which a genuine injected bug was not exercised by the verifier's test suite, so the episode would have been recorded as a success; the tests were strengthened until the faults were live. This is the point of executing the code rather than asserting the fault by inspection.

Trace format. Each trace contains the issue, the coordination plan, and an event log with stable event identifiers: A and B agreeing the contract (`e1`, `e2`); A reading the module (`e3`); A's unified diff (`e4`); A's hand-off message (`e5`); B's test file (`e6`); and the verbatim pytest output (`e7`). The hand-off message is fixed across all traces and is deliberately uninformative ("Ran it locally on the examples in the issue; looks right to me"), so it never announces a signature change; separating I from X requires comparing the diff against the contract in `e1`. The *artifacts-only* condition deletes `e1`, `e2` and `e5`, leaving the issue, the diff, the test and the traceback.

Table 3: **Measured attribution channels.** Four LLMs label n=96 coordination failures per condition, on traces whose true cause is fixed by construction. $\sigma_{\min}(\hat{M})$ is the statistic that Theorem 1 turns into an estimation rate; it is 0 exactly when attribution is uninformative. Recall is the diagonal of $\hat{M}$. *Artifacts only* removes the two inter-agent messages from the trace, leaving the issue, the diff, the test and the traceback. Brackets are 95% cluster bootstrap intervals over the 16 scenarios.

Trace	Labeller	Accuracy	$\sigma_{\min}(\hat{M})$	α_{eff}	Recall by cause		
					I	T	X
Full trace	GPT-4.1-mini	0.64 [0.55, 0.71]	0.28 [0.13, 0.42]	0.52	1.00	0.59	0.31
	GPT-4.1	0.65 [0.58, 0.70]	0.14 [0.05, 0.25]	0.43	1.00	0.75	0.19
	GPT-5-mini	0.73 [0.67, 0.79]	0.25 [0.12, 0.40]	0.50	0.97	0.91	0.31
	GPT-5	0.84 [0.77, 0.91]	0.53 [0.36, 0.70]	0.68	0.94	1.00	0.59
Artifacts only	GPT-4.1-mini	0.59 [0.51, 0.67]	0.19 [0.08, 0.30]	0.46	1.00	0.56	0.22
	GPT-4.1	0.67 [0.58, 0.74]	0.26 [0.15, 0.39]	0.51	1.00	0.69	0.31
	GPT-5-mini	0.68 [0.60, 0.75]	0.20 [0.07, 0.34]	0.46	1.00	0.78	0.25
	GPT-5	0.74 [0.69, 0.79]	0.22 [0.09, 0.37]	0.48	0.97	0.97	0.28
uninformative	*chance*	0.33	0.00	0.33	0.33	0.33	0.33

Table 4: **The controlled testbed driven by measured attribution channels.** Each column replaces the idealised symmetric channel of Table 1 with the $\hat{M}$ measured from that model (Table 3); the filter is given the same $\hat{M}$. Utility per episode over 50 seeds. Δ_{attr} is the paired gain of LPC over the same coordinator with attribution switched off, with a 95% bootstrap interval.

Method	GPT-4.1-mini	GPT-4.1	GPT-5-mini	GPT-5
$\sigma_{\min}(\hat{M})$	0.28	0.14	0.25	0.53
Reputation ranking	2.87	2.87	2.87	2.87
Plan-level Thompson sampling	3.78	3.78	3.78	3.78
LPC − semantic attribution	4.50	4.50	4.50	4.50
LPC (full)	4.60	4.68	4.73	4.75
Δ_{attr}	+0.10 [+0.01, +0.19]	+0.18 [+0.10, +0.25]	+0.23 [+0.15, +0.32]	+0.25 [+0.16, +0.34]
Oracle (true parameters)	5.55	5.55	5.55	5.55

Labeling. Four models (GPT-4.1-mini, GPT-4.1, GPT-5-mini and GPT-5) label each trace under both richness conditions, giving $4 \times 96 \times 2 = 768$ labels. The prompt states the three-cause taxonomy in the wording of Section 3.1 and asks for a cause, the event ids that justify it (interface contract C3) and a one-sentence rationale, as JSON. It says nothing about how the corpus was built and nothing about the prior over causes, which is uniform by construction but not disclosed. Decoding is deterministic where the API allows it. No label was unparseable and none was discarded.

Estimation. $\hat{M}$ is the column-normalized confusion matrix, $\sigma_{\min}(\hat{M})$ its smallest singular value, and $\alpha_{\text{eff}} = (2\sigma_{\min} + 1)/3$ the accuracy of the symmetric channel with the same $\sigma_{\min}$, by Theorem 1(iii). Intervals are 95% percentile intervals from a cluster bootstrap over the 16 scenarios with 4000 replicates; clustering on the scenario rather than the trace is the conservative choice, since the two variants of a cause within a scenario share a module and a contract. Intervals on $\sigma_{\min}$ are correspondingly wide; the claim they support is that $\sigma_{\min} > 0$, which holds for all eight conditions.

Feeding the measured channel back into the testbed. Table 4 re-runs the controlled experiment of Section 6 with the idealized symmetric channel replaced by each measured $\hat{M}$: the environment emits labels from $\hat{M}$, and the filter is given the same $\hat{M}$, so this is the correctly-specified case with a realistic, asymmetric channel. All other parameters are unchanged, and the 50 test seeds are the same.

What this study does not establish. It measures one component of LPC in isolation. It does not measure LLM-PROPOSE or LLM-ROLLOUT, it does not run a coordinator over real agents, and it does not report a resolve rate. Appendix I pre-registers the end-to-end study that would.

G OUTLOOK: WHERE THE MEASUREMENT SAYS EFFORT SHOULD GO

Because recall of X is the binding constraint on $\sigma_{\min}(\hat{M})$ (Table 3), attribution quality improves most from evidence that makes agreements machine-checkable, such as typed commitments and interface contracts logged as artifacts rather than as prose. That is a change to what agents *record*, not to how well they reason. The richness ablation supports this directly: deleting the two messages that record the agreement costs three of the four models recall on X, and costs the strongest model the most, so the information that separates X from I is carried by that record and is currently recovered from informal prose. A coordinator that requires agents to emit the contract in a typed form converts that extraction problem into a lookup, and the resulting channel should be closer to the identity than any of the eight we measured. Beyond this, richer belief models and self-interested agents fit the same interface (Appendix E).

H EXPERIMENTAL DETAILS AND ADDITIONAL RESULTS

Simulator. Code is provided in the supplementary material (`sim/lpc_sim.py`). Every environment draws its parameters, task contexts and a $T \times 7$ matrix of uniform random numbers from its seed. The draws drive implementation, alarm, replacement implementation, verification, third stage, attribution correctness and the attribution-confusion choice, so all methods face identical randomness whenever they make identical choices. Compatibility is symmetric unless stated otherwise, and the non-identifiability construction in Theorem 1 respects this symmetry.

Belief priors and baselines. Competences start at $\mathrm{Beta}(2,1)$, compatibilities at $\mathrm{Beta}(6,1)$ (most pairs compatible) and solo overload at $\mathrm{Beta}(2,1)$; all learning methods use these priors. The no-attribution ablation uses the same factorized model with uninformative labels ($\alpha = 1/3$ in (2)). Replaced implementers receive a fractional failure count of π (Proposition 3). The baselines are configured as follows:

- *Reputation ranking* keeps one pooled Beta posterior per agent across roles and contexts, updated from outcomes, and teams the top two agents, with the first as implementer.
- *Individual trust + attribution* uses role- and context-specific posteriors with a pooled integration rate, selects greedily, allows solo plans, and never re-coordinates.
- *Plan-level Thompson sampling* keeps a $\mathrm{Beta}(1,1)$ posterior per (plan, context) with the same forgetting factor, and does not re-coordinate.

Full results and ablations. Table 5 extends Table 1 with all ablations, the fraction of solo episodes and the re-coordination rate. Figure 7 shows adaptation to capability drift and robustness to a misspecified attribution accuracy.

Hyperparameter selection. On development seeds 1000 to 1011 we compared $T \in \{600, 1500\}$ and $\gamma \in \{0.97, 0.99, 0.995, 1\}$ (script `sim/dev_explore.py`). We fixed $T = 1500$, where learning has converged enough to separate ablations, and $\gamma = 0.99$. Test seeds are 0 to 49 for Table 1 and 0 to 29 for the sweeps.

Value of attribution versus population size. Paired gain from attribution (full LPC minus the no-attribution ablation), in utility per episode with 95% bootstrap intervals: $N = 4$: $+0.22$ $[0.12, 0.34]$; $N = 6$: $+0.23$ $[0.13, 0.33]$; $N = 10$: $+0.22$ $[0.09, 0.35]$; $N = 14$: $+0.21$ $[0.09, 0.32]$. With $N = 10$, utility is flat while the labels are near-uninformative and then increases with attribution accuracy α: 4.79 at $\alpha = 1/3$, 4.78 at 0.5, 4.99 at 0.7, 5.00 at 0.9, and 5.10 at 1.0. The value at $\alpha = 1/3$ coincides with the no-attribution ablation (4.79), as it must, since a filter told that labels are uninformative ignores them.

Table 5: Full results in the controlled testbed, including ablations (N=6, T=1500, drift at $T/2$, α=0.8; 50 seeds). Utility per episode with 95% bootstrap CI; Δ is the paired difference from LPC.

Method	Utility ↑	Δ vs. LPC	Success	Solo / Re-coord. (%)
Random plan	2.26 [2.15, 2.36]	-2.46 [-2.57, -2.35]	0.409	16.8 / 0.0
Reputation ranking	2.87 [2.60, 3.15]	-1.84 [-2.11, -1.58]	0.487	0.0 / 0.0
Individual trust + attribution	3.79 [3.56, 4.03]	-0.92 [-1.12, -0.72]	0.541	38.6 / 0.0
Plan-level Thompson sampling	3.78 [3.64, 3.92]	-0.93 [-1.00, -0.86]	0.562	15.8 / 0.0
LPC (full)	4.71 [4.58, 4.85]	n/a	0.685	15.6 / 14.4
– semantic attribution	4.50 [4.36, 4.64]	-0.21 [-0.29, -0.13]	0.658	26.3 / 17.2
– compatibility	4.50 [4.34, 4.65]	-0.21 [-0.30, -0.13]	0.646	33.4 / 14.7
– exploration (greedy)	4.28 [4.07, 4.48]	-0.44 [-0.64, -0.23]	0.630	18.8 / 10.5
– re-coordination	4.51 [4.36, 4.65]	-0.21 [-0.27, -0.14]	0.634	16.5 / 0.0
re-coordinate on every alarm	4.61 [4.49, 4.74]	-0.10 [-0.15, -0.06]	0.683	18.0 / 19.9
– solo option	4.73 [4.58, 4.87]	+0.01 [-0.08, +0.10]	0.701	0.0 / 14.3
– forgetting	4.58 [4.41, 4.74]	-0.13 [-0.21, -0.06]	0.670	18.9 / 15.5
Oracle (true parameters)	5.55 [5.43, 5.67]	+0.83 [+0.78, +0.89]	0.772	4.5 / 10.8

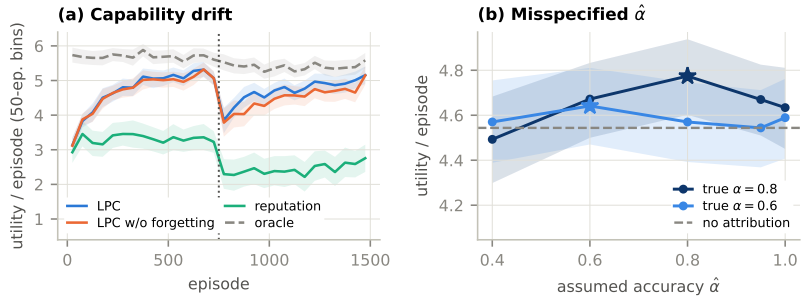


Figure 7: **Drift and misspecified attribution** (95% bootstrap bands). (a) Utility per episode (50-episode bins) around the silent capability drop (dotted). Reputation ranking never recovers; LPC recovers within about 100 episodes, and faster with forgetting. (b) LPC with an assumed attribution accuracy $\hat{\alpha}$. Stars mark the true value, and the dashed line is LPC without attribution.

Table 6: Robustness to environment misspecification (utility per episode, 30 seeds). The LPC belief model assumes symmetric compatibility and no hidden task difficulty. *Asymmetric k*: $k_{ij} \neq k_{ji}$. *Difficulty shocks*: on 30% of tasks every stage probability is multiplied by 0.75, unobserved. Last row: share of the reputation-to-oracle gap closed by LPC.

Method	Well-specified	Asymmetric k	Difficulty shocks	Both
Reputation ranking	3.04	3.09	2.06	2.19
Individual trust + attribution	3.87	3.92	2.99	2.83
Plan-level Thompson sampling	3.79	3.87	2.49	2.55
LPC	4.77	4.74	3.30	3.35
Oracle	5.54	5.56	4.20	4.22
Gap closed by LPC	69%	67%	58%	57%

Misspecified attribution accuracy. With true $\alpha = 0.8$, the correctly specified filter scores 4.77. Assuming $\hat{\alpha} \in \{0.6, 0.95, 1.0\}$ gives 4.67, 4.67 and 4.63; dropping attribution gives 4.54; underestimating with $\hat{\alpha} = 0.4$ gives 4.49. With weak attribution (true $\alpha = 0.6$), all settings lie within each other's confidence intervals (4.54 to 4.64). In practice $\hat{\alpha}$ should be estimated on audited traces and, if uncertain, erred upward.

Environment misspecification. Table 6 reports the stress tests summarized in Section 6.

Table 7: Re-coordination rules across re-coordination costs c_r (utility per episode, 30 seeds). Best fixed τ is chosen in hindsight per cost.

c_r	never	every alarm (τ=0)	τ=0.7	best fixed τ (hindsight)	$\tau^\star$ (LPC)
0.5	4.49	4.74	4.73	4.76 (τ=0.5)	4.79
2	4.49	4.44	4.63	4.63 (τ=0.7)	4.52
4	4.49	4.04	4.50	4.54 (τ=0.8)	4.54
mean	n/a	n/a	4.62	n/a	4.62

Belief-triggered re-coordination. The best fixed posterior threshold moves with the re-coordination cost, from $\tau = 0.5$ at $c_r = 0.5$ to $\tau = 0.8$ at $c_r = 4$ (Table 7). The belief-dependent $\tau^\star$ needs no tuning, is best at $c_r \in \{0.5, 4\}$, and matches the best single fixed threshold averaged over costs (4.62). At $c_r = 2$ it trails the per-cost hindsight optimum (4.52 versus 4.63), a gap consistent with the plug-in loss of Corollary 1.

Full re-coordination sweep. Utility per episode for every fixed posterior threshold τ:

c_r	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	never	$\tau^\star$
0.5	4.74	4.74	4.74	4.74	4.74	4.76	4.74	4.73	4.57	4.49	4.49	4.79
2	4.44	4.44	4.44	4.44	4.45	4.52	4.57	4.63	4.56	4.48	4.49	4.52
4	4.04	4.04	4.04	4.05	4.06	4.20	4.33	4.50	4.54	4.48	4.49	4.54

Calibration. Mean Brier score of the predicted success probability of the selected plan, over episodes without re-coordination: LPC 0.202; without attribution 0.215; without compatibility 0.210; individual trust with attribution 0.229. Absolute calibration is limited by the intrinsic outcome variance of Bernoulli tasks.

I Pre-registered Protocol for the End-to-End Study

Appendix F measures the attribution channel, which is the quantity Theorem 1 depends on. It does not measure end-to-end coordination gain over real agents on a real benchmark, and no claim in this paper does. This appendix pre-registers that study so the design is fixed in advance of the result. We do not run all 652 CooperBench tasks (Khatua et al., 2026): a full sweep would buy precision we do not need and would confound attribution quality with agent quality.

Agents and tasks. The agents are four pinned (model, prompt, tool) configurations: two base models crossed with implementation-specialized and test-specialized prompts. Tasks are stratified by repository and by the difficulty tier reported with CooperBench, and ordered chronologically within a repository so that retrieved histories never contain test solutions. Additional histories for fitting beliefs come from SWE-smith (Yang et al., 2025). Workers emit typed commitments (files, symbols, interface contracts, promised tests), and checkpoints run interface and unit tests in sandboxed containers (Yang et al., 2024; Wang et al., 2025).

Replicating the channel measurement in the wild. The first step repeats Appendix F on repository-scale failures, where the true cause is contestable and must be adjudicated rather than constructed. We label 150 *failed* traces, sampled to balance the three causes, against independent human labels with a third adjudicator for disagreements, and report $\hat{M}$, $\sigma_{\min}(\hat{M})$ and a bootstrap interval, together with the inter-annotator agreement that bounds how much of the residual is adjudication noise. At $n = 150$ with three causes, the standard error on a symmetric-channel $\hat{\alpha}$ is about 0.065 and on $\sigma_{\min}$ about 0.10, which separates any $\hat{\alpha} \gtrsim 0.55$ from the degenerate $\alpha = 1/3$ at the 2σ level. The primary comparison is against Table 3: we expect lower $\sigma_{\min}$ and predict that the $\mathsf{X} \to \mathsf{I}$ bias persists. This step reuses traces logged from solo and cooperative runs and requires no coordinator. We additionally report $\hat{M}$ broken out per agent configuration and per language, because a bias that is invisible in aggregate still enters the estimates through M (see the Ethics Statement).

Coordination gain. A paired design over 120 tasks, 10 per repository, with each task run under every method and identical system budgets. Powering for the paired McNemar test at $\alpha = 0.05$, 120 tasks give roughly 80% power against a 15 point difference in resolve rate under a discordance rate of 0.29; detecting 10 points would need about 220 tasks, which we regard as out of scope. We report resolve rate, tokens, wall-clock time and coordination gain relative to the strongest single agent, and we fix the primary comparison (LPC versus the strongest single agent) in advance to avoid selecting among baselines post hoc.

Secondary measurements.

(i) Identifiability of agent identity: the incremental predictive value of identity after conditioning on task features. If it is negligible, beliefs should be indexed by configuration features rather than names.

(ii) Calibration (Brier score, ECE) of rollout-based predictions of success and commitment violation.

(iii) Sensitivity of the coordination-gain result to the channel, using the $\hat{M}$ measured in the wild in place of the constructed-corpus channels of Table 3.

Baselines and statistics. Baselines are the strongest single agent; a fixed two-agent workflow; a manager LLM with current-episode context; ETI-style partner profiles (Abdurahman et al., 2026); reputation ranking; and LPC with its ablations. We use paired McNemar tests and paired bootstrap intervals over tasks with Holm correction, and mixed-effects logistic regression with repository random effects.