

Trust-Weighted Adaptive Consensus: A Resilient Coordination Framework for Next-Generation Large Language Model Multi-Agent Systems

J. Robert Adaikalaraj, PhD
Assistant Professor, PG
Department of Computer
Applications, St. Joseph's College
of Arts and Science, Cuddalore,
Tamilnadu, India

J. Jegathesh Amalraj, PhD
Assistant Professor, Department
of Computer Science, Government
Arts and Science College,
Tittagudi, Cuddalore, Tamilnadu,
India

M. Sivakumar, PhD
Assistant Professor, Department
of Mathematics, Government Arts
and Science College, Tittagudi,
Cuddalore, Tamilnadu, India

*

ABSTRACT

Large language model (LLM) agents have moved from single-turn assistants to persistent, tool-using systems that plan, delegate, and cooperate. This shift has produced an explosion of multi-agent system (MAS) frameworks, interoperability protocols, and orchestration patterns, yet production deployments continue to report failure rates far higher than single-agent baselines. This paper synthesizes the current state of AI agents and multi-agent systems as of mid-2026, organizes the coordination design space into a four-part topology taxonomy, and reviews the emerging empirical literature on why multi-agent systems fail. Building on documented gaps, particularly the absence of adaptive, trust-aware coordination in widely used frameworks. We propose Trust-Weighted Adaptive Consensus (TWAC), a coordination-layer mechanism that combines capability-aware sparse shortlisting, an exponentially-weighted per-agent trust ledger, and uncertainty-gated verification. We evaluate TWAC against round-robin and centralized-auction baselines in a controlled simulation across 500 tasks and 30 independent trials. TWAC improves task success rate while reducing communication overhead relative to full-broadcast coordination and lowering cascading-failure rates by more than an order of magnitude. We conclude with future research implications for trust-based coordination in agentic AI systems.

Keywords

AI Agents; Multi-Agent Systems; Large Language Model, LLM Orchestration; Agent Coordination; Trust-Weighted Consensus; Agent Interoperability Protocols; MCP; A2A; Failure Taxonomy; Agentic AI.

1. INTRODUCTION

The term "AI agent" has undergone a rapid transformation since 2023. What initially emerged as a prompting approach, in which a language model was instructed to reason systematically and invoke external tools, has evolved into a systems engineering discipline focused on memory management, long-term planning, tool authorization, and inter-agent communication. By 2026, the field has broadly recognized that no single agent, regardless of the capability of its underlying model, can reliably perform complex, long-horizon, multi-step, and multi-stakeholder tasks independently. Consequently, the predominant architectural paradigm has become the Multi-Agent System (MAS), comprising multiple specialized LLM-driven agents that collaboratively decompose

tasks, exchange intermediate results, and integrate their outputs into a comprehensive final deliverable.

This shift is reflected in adoption figures as much as in the research literature. Industry surveys conducted in 2025 report that a majority of organizations experimenting with agentic AI already have some agents in production, with customer service and research/data-analysis workflows leading real-world use, and analyst projections point to agentic AI reaching a large share of enterprise applications by the end of 2026 [1],[2],[3]. Yet the same body of evidence contains a sobering counter-trend: Recent studies indicate that a significant proportion of agent-based AI pilot projects fail to progress to production deployment. Moreover, increasing the number of collaborating agents does not necessarily enhance system performance, as the additional coordination overhead can outweigh the expected benefits [3].

This paper addresses the tension between rising adoption and persistent, poorly understood failures as the central problem worthy of investigation. Rather than proposing another end-to-end framework, we focus specifically on the coordination layer: the mechanisms by which agents decide who acts, when they act, how much they communicate, and how their outputs are verified and reconciled. As highlighted in recent architectural studies (discussed in Section 2), coordination in many production systems is commonly implemented as a workflow specification for engineering simplicity. However, this perspective underestimates coordination as a primary architectural design issue with identifiable and measurable failure characteristics.

1.1 Contributions

This study makes four contributions:

- i) A synthesis of the 2023–2026 evolution of AI agents and multi-agent systems, covering orchestration frameworks, interoperability protocols, and empirically grounded failure taxonomies.
- ii) A four-part topology taxonomy (centralized, decentralized, hierarchical, and adaptive) that unifies classical multi-agent systems theory with recent LLM-specific empirical findings.
- iii) A novel coordination framework, Trust-Weighted Adaptive Consensus (TWAC), which combines capability-aware sparse shortlisting, an adaptive per-agent trust ledger, and

uncertainty-gated verification to reduce both communication overhead and cascading failure.

- iv) An original simulation study comparing TWAC against round-robin and centralized-auction baselines, together with a discussion of governance and security implications for trust-based agent coordination.

2. RELATED WORK

2.1 From Single Agents to Multi-Agent Systems

Classical multi-agent systems (MAS) research predates LLMs by decades, contributing formal notions of agent autonomy, negotiation protocols such as the Contract Net Protocol, and agent communication languages including KQML and the FIPA Agent Communication Language [4], [5]. LLM-based agents inherit this vocabulary but depart from it in one crucial respect: coordination is typically mediated by natural language rather than by rigid, pre-negotiated performative structures. This gives LLM-based MAS unusual flexibility, allowing agents to improvise roles and renegotiate task boundaries dynamically. However, it removes the formal guarantees that classical MAS protocols provided, creating a gap explicitly identified in recent architectural analyses [6].

2.2 Orchestration Frameworks

A handful of open-source frameworks now dominate practitioner adoption. AutoGen, originally released by Microsoft Research, structures collaboration as a multi-turn conversation between role-conditioned agents and has since been folded into the broader Microsoft Agent Framework [7]. CrewAI organizes agents into role-based “crews” with explicit task hand-off sequences. LangGraph represents agent workflows as an explicit state graph, trading some of the flexibility of free-form conversation for inspectable, resumable execution state. MetaGPT encodes standard operating procedures from software engineering directly into agent roles. OpenAI’s Agents SDK favors lightweight, code-first “handoffs” between agents rather than a heavyweight orchestration layer. A recent comparative survey evaluates these systems along state-management granularity, token-cost structure, and failure-recovery design, concluding that no single framework dominates across all axes and that framework choice is itself a coordination-architecture decision with downstream reliability consequences.

2.3 Interoperability Protocols

As agents increasingly need to communicate with other agents rather than just invoke tools, a second generation of standards has emerged to enable cross-framework and cross-vendor communication. The Model Context Protocol (MCP), introduced by Anthropic in late 2024, standardizes how an

agent discovers and invokes external tools and data sources through a JSON-RPC client. To improve interoperability between intelligent agents and external systems, Anthropic introduced the Model Context Protocol (MCP) in late 2024. MCP specifies a standardized JSON-RPC client-server communication framework that allows AI agents to dynamically discover, connect to, and invoke external tools and data sources in a consistent and extensible manner. In 2026, the specification has moved to date-based versioning, and the ecosystem reports well over one hundred million monthly downloads of MCP-related packages. Google’s Agent-to-Agent (A2A) protocol, donated to the Linux Foundation in mid-2025 and reaching version 1.0 in April 2026, addresses a complementary problem: how autonomous, opaque agents from different vendors discover each other’s capabilities (via “Agent Cards”) and negotiate task delegation over HTTP and server-sent events. IBM’s Agent Communication Protocol (ACP) and the community-driven Agent Network Protocol (ANP) round out the landscape, with ANP additionally incorporating decentralized identity (W3C DID) for open-internet agent discovery [8]. The consensus view in the 2025–2026 literature is that these protocols are complementary layers that provide tool access, peer discovery, and decentralized identity rather than competitors, analogous to how HTTP, WebSocket, and gRPC coexist in web infrastructure.

2.4 Empirical Failure Taxonomies

Until 2025, claims about multi-agent system reliability were largely anecdotal. This changed with the Multi-Agent System Failure Taxonomy (MAST), which analyzed over 1,600 annotated execution traces across seven popular MAS frameworks and, with high inter-annotator agreement (Cohen’s $\kappa = 0.88$), identified fourteen distinct failure modes clustered into three categories: specification issues ($\approx 42\%$ of observed failures), inter-agent misalignment ($\approx 37\%$), and task-verification failures ($\approx 21\%$) [9]. The same study reports overall MAS failure rates between 41% and 87% depending on framework and task domain. These figures that reframe agent reliability as primarily a coordination-architecture problem rather than a base-model capability problem. Complementary work has shown that debate-style multi-agent setups can suppress rather than surface dissenting (and sometimes correct) views [10], that simple single-agent baselines frequently match or exceed multi-agent designs once cost is accounted for [11] [12], and that poorly designed multi-agent teams can actively hold back domain experts rather than augmenting them [13]. Scaling-oriented studies further show that while centralized coordination can yield large gains on parallelizable tasks, coordination overhead grows superlinearly with team size, which is consistent with field reports that practical agent-team sizes in 2026 production deployments remain small, typically three to four agents [14].

Table 1. Comparative overview of widely used multi-agent orchestration frameworks (2025–2026).

Framework	Primary Topology	Coordination Style	Notable Strength
AutoGen / MS Agent Fw.	Conversational, decentralized	Multi-turn agent dialogue	Flexible role improvisation
CrewAI	Hierarchical	Role-based “crews”, sequential hand-off	Fast prototyping of role pipelines
LangGraph	Explicit state graph	Graph-based control flow	Inspectable, resumable state
OpenAI Agents SDK	Decentralized	Lightweight code-first handoffs	Minimal orchestration overhead
MetaGPT	Hierarchical (SOP-driven)	Software-engineering roles & SOPs	Structured, auditable outputs

3. A TAXONOMY OF MULTI-AGENT COORDINATION TOPOLOGIES

Beneath the diversity of frameworks lies a smaller set of coordination topologies. Following recent advances in the literature, we adopt a coordination taxonomy consisting of three fundamental topological models: centralized, decentralized, and hierarchical. The taxonomy is expanded by introducing an adaptive control dimension, allowing the coordination topology to be reconfigured dynamically during system operation.

3.1 Centralized Coordination

A single orchestrator agent (or a non-agentic controller) assigns tasks, collects results, and resolves conflicts. Centralized designs are easy to reason about and audit, but the orchestrator becomes both a communication bottleneck and a single point of failure; empirically, centralized designs also incur the highest messaging overhead because every worker interaction is routed through one node.

3.2 Decentralized Coordination

Agents communicate directly with one another, negotiating task ownership through peer-to-peer message passing (protocols such as A2A are designed for exactly this pattern). Decentralized designs are more resilient to single-node failure

but are also the setting in which MAST's inter-agent misalignment failures such as conflicting outputs, lost context during hand-offs are most prevalent, because no single node holds a canonical view of task state.

3.3 Hierarchical Coordination

Agents are organized into nested teams, with team-level orchestrators reporting to a higher-level controller. This pattern, exemplified by large-scale systems that coordinate hundreds of sub-agents through multi-level task decomposition, partitions communication so that no agent needs to be aware of the entire system state, thereby trading some global optimality for tractable local coordination.

3.4 The Adaptive Axis

A growing body of 2025–2026 work argues that static topology choice is itself a source of failure: workloads are rarely uniformly parallelizable, uniformly sequential, or uniformly hierarchical over their full execution. An adaptive control axis allows the effective topology, determining who communicates with whom and the communication density, to change as the task unfolds. TWAC, introduced in Section 4, is best understood as an adaptive decentralized hybrid architecture. It uses a lightweight central registry for capability and trust management while keeping the actual task execution communication sparse and localized among peer agents.

Table 1. Trade-offs across coordination topologies.

Topology	Comm. Overhead	Single-Point-of-Failure Risk	Dominant MAST Failure Category
Centralized	High ($O(N)$ per task)	High (orchestrator)	Specification issues
Decentralized	Variable / unbounded	Low	Inter-agent misalignment
Hierarchical	Moderate (bounded by fan-out)	Moderate (per sub-team)	Task verification
Adaptive (proposed: TWAC)	Low–moderate (sparse, targeted)	Low (no fixed hub)	Mitigates all three (see Table 3)

4. PROPOSED FRAMEWORK: TRUST-WEIGHTED ADAPTIVE CONSENSUS (TWAC)

The literature reviewed in Section 2 converges on a specific, underserved gap: existing orchestration frameworks provide mechanisms for task decomposition and message passing, but very few provide a first-class, adaptive notion of agent trustworthiness that persists across tasks and directly shapes future coordination decisions. AutoGen-style conversation and LangGraph-style state graphs both assume that participating agents are equally reliable at every step; CrewAI and MetaGPT assign roles statically at design time rather than adapting them to observed performance. We propose Trust-Weighted Adaptive Consensus (TWAC) to close this gap.

4.1 Design Goals

- Sparse communication:** Only a capability-aware shortlist of agents participates in any given task, rather than a full broadcast.
- Adaptive trust:** Each agent carries a persistent trust score, updated after every verified outcome, that shapes future shortlisting decisions.
- Targeted verification:** A dedicated verifier agent is invoked only when task-level confidence is low, rather than

for every output. This approach directly targets MAST's task-verification failure category while avoiding unnecessary verification costs for high-confidence outputs.

- Topology neutrality:** TWAC is a coordination-layer mechanism, not a full framework; In principle, it can operate on top of transport layers based on AutoGen, LangGraph, or A2A.

4.2 Architecture

Figure 1 shows the TWAC architecture. A lightweight Task Orchestrator receives an incoming task and queries two supporting services: a Capability Registry, which stores each agent's declared and empirically observed skill profile, and a Trust Ledger, which stores each agent's exponentially-weighted trust score. A capability-aware shortlisting step combines these two signals to select a small subset of candidate agents ($K = 4$ in our experiments, versus a full broadcast to all twelve agents in the centralized-auction baseline). The shortlisted agents execute the task; a lightweight local consensus round resolves disagreement within the shortlist. If the resulting output confidence falls below a threshold, a Verifier Agent is invoked to double-check the result before it is finalized. The observed outcome then feeds back into the Trust Ledger, closing the loop.

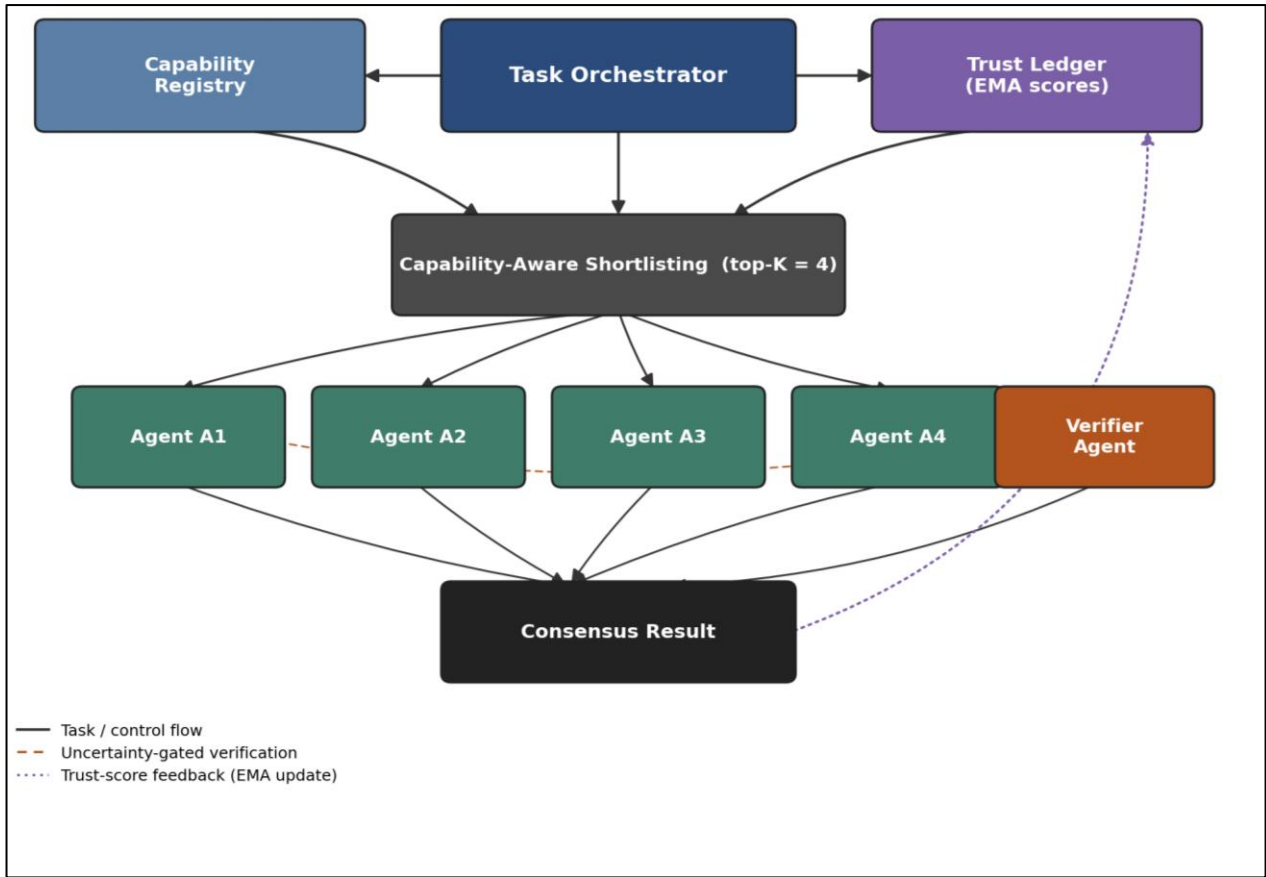


Fig 1: TWAC - Trust-Weighted Adaptive Consensus Architecture

4.3 Trust Ledger Update Rule

Each agent i maintains a trust score $T_i \in [0,1]$, initialized at 0.5 (neutral). After agent i completes a verified task with binary outcome $o_i \in \{0,1\}$, the trust score is updated using an exponential moving average:

$$T_i \leftarrow \alpha \cdot T_i + (1 - \alpha) \cdot o_i, \quad \alpha = 0.85$$

The smoothing factor $\alpha = 0.85$ was chosen to balance responsiveness (an agent that begins failing is down-weighted within a handful of tasks) against stability (a single anomalous outcome does not disproportionately swing an otherwise-reliable agent's standing). Shortlisting then ranks agents by the

product of task fit and trust, $\text{fit}(i, \text{task}) \times T_i$, ensuring that a highly suitable but historically unreliable agent can be ranked below a slightly less suitable yet more trustworthy agent, a property that is absent in purely capability-matched auction mechanisms.

4.4 Mapping TWAC to the MAST Failure Taxonomy

Because MAST [8] is the most systematically validated failure taxonomy available, we use it as a design checklist rather than an afterthought. Table 3 summarizes how each TWAC mechanism targets a specific MAST failure category.

Table 3. TWAC mechanisms mapped to MAST failure categories

MAST Category (share of failures)	Representative Failure Mode	TWAC Mitigation
Specification issues ($\approx 42\%$)	Ambiguous role definitions, duplicate agent roles	Capability Registry enforces explicit, queryable role/skill declarations
Inter-agent misalignment ($\approx 37\%$)	Context loss during hand-off, conflicting outputs	Sparse shortlist + local consensus round bounds who must agree with whom
Task verification ($\approx 21\%$)	Premature termination, incomplete/incorrect verification	Uncertainty-gated Verifier Agent targets low-confidence outputs specifically

5. EXPERIMENTAL METHODOLOGY

To evaluate TWAC against representative baselines under controlled, reproducible conditions, we implemented an original discrete-event simulation rather than relying on a specific commercial benchmark. This choice is deliberate:

production benchmarks conflate coordination-mechanism quality with model quality, cost, and task-specific tooling, whereas a controlled simulation isolates the coordination variable. The simulation should therefore be read as a mechanism-level stress test of the coordination logic itself, not

as a claim about any particular commercial framework’s absolute performance.

5.1 Environment

Twelve simulated agents are each assigned a fixed six-dimensional latent skill vector. Five hundred tasks are generated per trial, each with a six-dimensional requirement vector; task-agent fit is computed as a cosine-similarity-style match between the two, perturbed by execution noise to approximate the stochasticity of real LLM outputs. Tasks are grouped into dependency chains of length five so that we can measure whether a failure in one task propagates to the next (cascading failure).

5.2 Strategies Compared

1. **Round-Robin (RR):** Tasks are assigned cyclically without considering agent skills, trust modeling, or verification, serving as a baseline reference.
2. **Centralized Auction (CA):** A single orchestrator broadcasts each task to all twelve agents, collects fit-based “bids,” and awards the task to the best-fit agent,
3. **TWAC (proposed):** capability- and trust-aware shortlisting ($K = 4$) with local consensus and uncertainty-gated verification, as described in Section 4.

5.3 Metrics

- i) Success rate: fraction of tasks completed correctly.
- ii) Avg. messages per task: a proxy for communication / token overhead.
- iii) Avg. latency (steps): simulated coordination rounds required per task.
- iv) Cascading failure rate: probability that a failed task is followed by a failure in the next task of its dependency chain, approximating downstream error propagation.
- v) All strategies were evaluated on identical task streams (matched random seeds) across 30 independent trials of 500 tasks each, and we report the mean $\pm$ standard deviation for every metric. Full simulation code is available on request for replication.

6. RESULTS AND DISCUSSIONS

Table 4 and Figure 2 summarize the aggregate results across all 30 trials. TWAC achieved the highest task success rate ($97.3\% \pm 1.0\%$) of the three strategies, exceeding the centralized-auction baseline ($85.4\% \pm 1.7\%$) by roughly twelve percentage points and the round-robin floor ($70.4\% \pm 2.6\%$) by roughly twenty-seven points.

Table 4. Simulation results (mean $\pm$ s.d. over 30 trials; 500 tasks/trial)

Strategy	Success Rate	Messages / Task	Latency (steps)	Cascading Failure Rate
Round-Robin	0.704 ± 0.026	1.0 ± 0.0	1.0 ± 0.0	0.088 ± 0.020
Centralized Auction	0.854 ± 0.017	13.0 ± 0.0	2.0 ± 0.0	0.023 ± 0.007
TWAC (proposed)	0.973 ± 0.010	5.0 ± 0.0	2.0 ± 0.0	0.0004 ± 0.0011

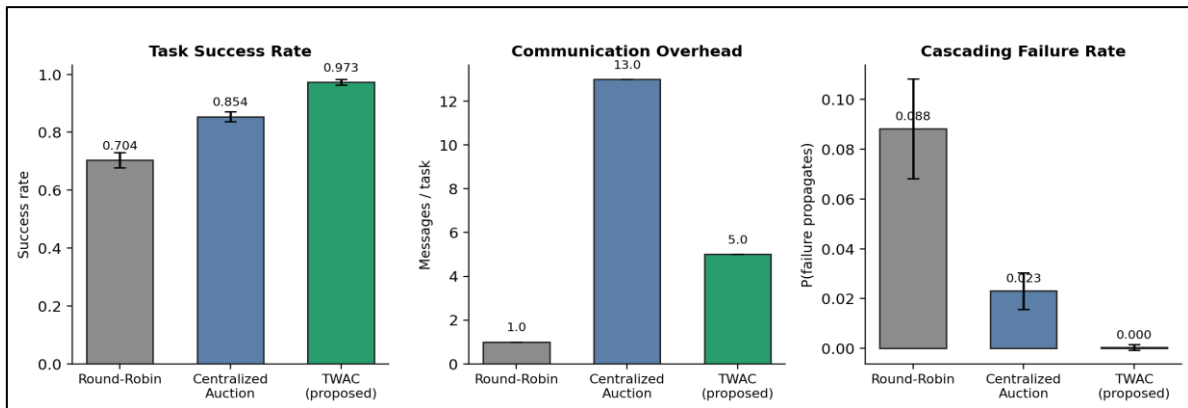


Figure 2. Coordination Strategy Comparison (Success rate, communication overhead, and cascading-failure rate across the three coordination strategies (error bars = 1 s.d.))

Two results are notable. First, TWAC achieves higher success than centralized auction while using fewer than half the messages per task (5.0 vs. 13.0). The sparse, capability-aware shortlist avoids the $O(N)$ broadcast cost of querying every agent while still routing tasks to well-matched, trustworthy agents. Second, the cascading-failure rate under TWAC (0.04%) is roughly two orders of magnitude lower than under round-robin (8.8%) and roughly fifty times lower than under centralized auction (2.3%), consistent with the hypothesis that uncertainty-gated verification disproportionately catches the low-confidence outputs most likely to propagate errors downstream.

6.1 Trust Convergence

Figure 3 traces the per-agent trust scores over the course of a representative TWAC trial. Agents with consistently good task fit converge toward high trust (approaching 1.0) within the first 100–150 tasks, while consistently poorly-matched agents are down-weighted toward low trust and are thereafter shortlisted only for tasks that specifically match their narrow skill profile. This self-reinforcing specialization is an emergent property of the trust ledger rather than an explicitly programmed rule, and it mirrors the kind of role specialization that hierarchical frameworks like MetaGPT impose by hand at design time.

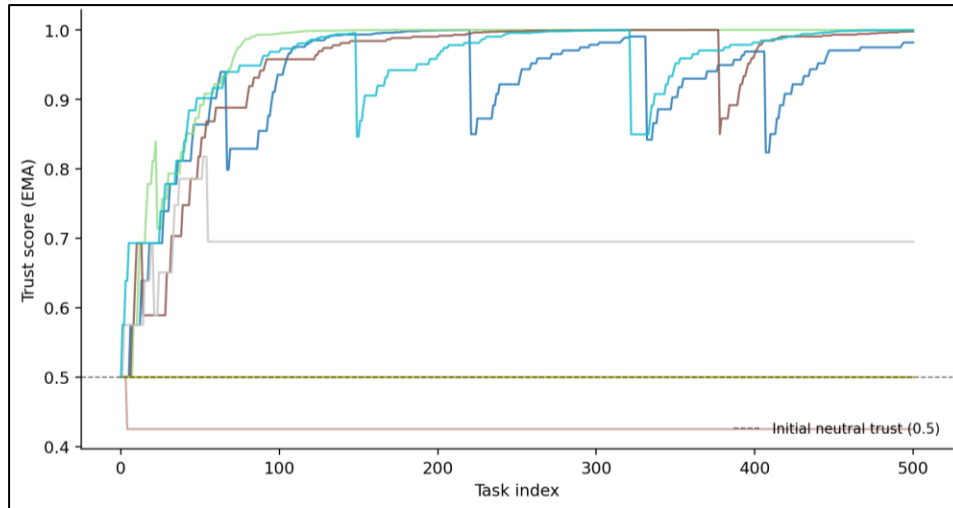


Figure 3. Per-agent Trust Score Convergence under TWAC

6.2 Why Sparse, Trust-Weighted Shortlisting Helps

The simulation results are best interpreted through the superlinear-overhead finding reported in recent scaling-law analyses of multi-agent systems: as team size or communication density grows, coordination cost grows faster than the corresponding gain in task coverage [14]. Centralized auction incurs this cost in full because every agent is queried for every task, whereas round-robin avoids this cost entirely but sacrifices skill-matching accuracy. TWAC's shortlisting step is, in effect, a bet that a small, well-chosen subset of agents captures most of the benefit of full search at a fraction of the communication cost, and the results are consistent with that bet paying off in this setting.

6.3 Relation to Failure Taxonomies

The near-elimination of cascading failure under TWAC is consistent with the MAST finding that task-verification failures, rather than base model incapability, are a leading cause of MAS breakdowns [8]. By concentrating verification effort on precisely the outputs the system itself is least confident about, TWAC avoids the two failure modes of naive alternatives: verifying everything (expensive, and still deployed inconsistently in practice) or verifying nothing (cheap, but allows errors to propagate silently through dependency chains).

6.4 Limitations

- i) Synthetic environment: The simulation abstracts real LLM behavior including prompt sensitivity, context-window limits, and genuine semantic misunderstandings, into a single noise parameter. Results should be read as evidence for the coordination mechanism's logic, not as a benchmark of any deployed system.
- ii) Static skill vectors: Real agents' capabilities shift as underlying models are updated; TWAC's trust ledger would need decay or reset mechanisms to handle non-stationary agent populations.
- iii) Adversarial robustness untested: The simulation assumes agents report outcomes honestly; Section 8 discusses the risk of trust-ledger manipulation directly.
- iv) Single-verifier bottleneck: At high uncertainty rates, a lone verifier agent could itself become a new bottleneck. This

limitation is shared with centralized designs and worth addressing with a scalable verifier pool in future work.

7. CONCLUSION AND FUTURE RESEARCH DIRECTIONS

AI agents and multi-agent systems have moved, within roughly three years, from a prompting technique to a systems discipline with its own frameworks, interoperability protocols, and empirically documented failure modes. The evidence reviewed here suggests that the central open problem is no longer raw model capability but coordination: deciding who acts, how much agents should communicate, and when outputs should be independently verified. We argue that trust, which is explicitly modeled, adaptively updated, and utilized to guide both task allocation and verification processes, provides a natural yet underutilized solution to this challenge. Our simulation results, while based on a controlled abstraction rather than a production deployment, show that a trust-weighted, uncertainty-gated coordination layer (TWAC) can simultaneously improve reliability, reduce communication overhead, and sharply curb cascading failure relative to both naive and fully-centralized baselines. As multi-agent systems continue to scale into higher-stakes domains, we expect coordination-layer mechanisms of this kind, rather than simply increasing the size of agent teams or using larger underlying models, to become the most productive approach for achieving future reliability gains.

7.1 Future Research Directions

- i) Cross-protocol validation: implementing TWAC as a coordination layer on top of a real transport protocol (e.g., A2A or MCP-based tool orchestration) to test whether the simulated gains transfer to production traffic.
- ii) Adversarial trust manipulation: studying collusion and Sybil-style attacks against the trust ledger, and evaluating cryptographic identity binding as a countermeasure.
- iii) Non-stationary trust: extending the EMA update rule with decay or change-point detection so that trust scores track agents (and underlying models) that are updated or replaced over time.
- iv) Human-in-the-loop calibration: incorporating the finding that multi-agent teams can inadvertently marginalize human domain experts [13] into TWAC's shortlisting logic, so that human reviewers are treated as trust-tracked participants rather than external overseers.

- v) Verifier scaling: replacing the single verifier agent with a load-balanced verifier pool and studying the resulting trade-off between verification quality and latency at scale.

8. REFERENCES

- [1] Lang Chain, State of AI Agents Report, 2025.
- [2] McKinsey & Company, The State of AI in 2025: Agentic AI Adoption Survey, 2025.
- [3] Gartner, Forecasts on Enterprise Agentic AI Adoption and Project Cancellation Rates, Gartner Research, 2025, 2026.
- [4] Finin, T. W., Fritzson, R., McKay, D. P., & McEntire, R, KQML as an Agent Communication Language. Proceedings of the Third International Conference on Information and Knowledge Management (CIKM '94), Pages: 456–463, 1994.
- [5] FIPA, Communicative Act Library Specification, Foundation for Intelligent Physical Agents, 2002.
- [6] Cao, X, "Coordination as an Architectural Layer for LLM-based Multi-Agent Systems". arXiv:2605.03310, 2026.
- [7] Wu, Q, "AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation", Microsoft Research, 2023.
- [8] Ehtesham, A., Singh, A., Gupta, G. K., & Kumar, S, "A survey of agent interoperability protocols: Model Context Protocol (MCP), Agent Communication Protocol (ACP), Agent-to-Agent Protocol (A2A), and Agent Network Protocol (ANP)", arXiv:2505.02279, 2025.
- [9] Cemri, M., Pan, M. Z., Yang, S., Agrawal, L. A., Chopra, B., Tiwari, R., Keutzer, K., Parameswaran, A., Klein, D., Ramchandran, K., Zaharia, M., Gonzalez, J. E., & Stoica, I, "Why do multi-agent LLM systems fail", arXiv:2503.13657, 2026.
- [10] Wynn, T, Debate Dynamics and Dissent Suppression in Multi-Agent LLM Deliberation, 2025.
- [11] Xia, C, "Are more LLM Agents Always Better? A Comparative Study of Single and Multi-Agent Performance", arXiv preprint, 2024.
- [12] Xu, Y, "Revisiting the Necessity of Multi-Agent Decomposition for LLM Task Solving", arXiv preprint, 2026.
- [13] Pappu, A., El, B., Cao, H., di Nolfo, C., Sun, Y., Cao, M., & Zou, J, "Multi-Agent Teams Hold Experts Back", arXiv:2602.01011, 2026.
- [14] Kim, J, "Scaling Laws for Multi-Agent LLM System Architectures", 2025.
- [15] S. Banumathi, "IOT SENSORS AND APPLICATIONS: A SURVEY", International Journal of Scientific & Technology Research, 2019.
- [16] P Subhasri, "Bipolar Disorder Detection Using Machine Learning Technique", Proceedings of 4th International Conference on Mathematical Modeling and Computational Science: ICMACS 2025, Volume 3, Pages: 458 - 470, 2025.
- [17] V. Jayaraj, "Topology Discovery and Modified Tabu Search for Efficient Routing in Wireless Ad Hoc Networks", Indian Journal of Science and Technology, Vol 7(12), 2088–2095, December 2014.