When Coordination Is Avoidable: A Monotonicity Analysis of Organizational Tasks

Harang Ju
Carey Business School, Johns Hopkins University
Baltimore, MD 21202
harang@jhu.edu

Abstract

Organizations devote substantial resources to coordination, yet which tasks actually require it for correctness remains unclear. The problem is acute in multi-agent AI systems, where coordination cost is directly measurable and can exceed the cost of the work itself. Distributed systems theory provides a precise criterion: coordination is required when a task specification is non-monotonic, meaning that as histories grow, new information can invalidate prior conclusions. Here we show that Thompson's classic taxonomy of interdependence maps to that criterion, yielding a decision rule for when coordination is required for correctness. We formalize the correspondence in a bridge theorem, apply the rule to 65 APQC workflows and (with a calibrated LLM) 13,417 O*NET tasks, and illustrate it in multi-agent AI simulations. Under our decompositions, 74% of workflows and 42% of O*NET tasks are monotonic, implying that up to 24–57% of coordination spending is unnecessary for correctness.

Keywords: coordination | monotonicity | organizational theory | multi-agent systems | coordination costs

Introduction

Coordination is a fundamental cost of organization. Meetings, approval chains, status updates, and cross-functional reviews all exist because organizations assume that coordination is necessary. Yet no formal criterion distinguishes the coordination that is necessary for correctness from the coordination that is not. The cost of this ambiguity is substantial and growing, especially with the recent proliferation of AI agents. Multi-agent AI systems, which inherit their coordination topology from the organizations they assist, now devote 40–60% of total compute to coordination cost [1], and the cost grows super-linearly with group size [2]. Because these systems make coordination costs directly countable in tokens, they also provide a clean empirical setting for testing theories of coordination necessity. The standard response is to optimize coordination by improving communication, streamlining processes, or adding coordination technology. Optimization, however, assumes that coordination is necessary. What if much of it is not?

Whether coordination is necessary depends on the structure of the task. Thompson's [3] interdependence taxonomy classifies tasks by the coordination demands their structure imposes. Pooled tasks contribute independently to a common outcome and need only uniform rules. Sequential tasks pass output from one unit to the next and require planning and scheduling. Reciprocal tasks require mutual adjustment among units, the most costly coordination form. Thompson observed that pooled and sequential interdependence dominate, with reciprocal interdependence concentrating at

critical integration points. This taxonomy remains foundational in organizational theory. Subsequent work has drawn sharper distinctions among task interdependence, agent interdependence, epistemic interdependence, and the fit between interdependence structures and coordination mechanisms [4, 5]. Here, we focus on one formal dimension: when coordination is required for correctness under a specified task representation.

Distributed computing contributes a formal criterion on this dimension. The CALM theorem (Consistency As Logical Monotonicity) proves that a computation can execute without coordination if and only if its specification is monotonic, meaning that as admissible histories expand, previously valid conclusions need not be retracted [6, 7]. Non-monotonic computations, where new information can invalidate prior conclusions, provably require coordination for correctness. The theorem has been applied extensively to databases [8] and recently unified with other coordination criteria over Lamport histories [9], but not to organizational task analysis. CALM governs correctness, not broader performance objectives. A monotonic task may still benefit from coordination for speed, stylistic consistency, or other goals, but correctness does not require it.

In this paper, we map Thompson's taxonomy to CALM's monotonicity criterion. The Thompson-CALM Bridge Theorem (Theorem 1) shows that pooled interdependence maps to monotone specifications and sequential interdependence maps to monotone specifications under causal ordering. The reciprocal case depends on feedback type. Reciprocal tasks become non-monotone when feedback retracts or constrains previously admissible outcomes, but can remain monotone when agents only add to one another's output. The Feedback Boundary (Proposition 1) extends the same distinction to sequential tasks with feedback, clarifying when feedback preserves monotonicity and when it introduces non-monotonic revision. The Coordination Tax (Equation 1) then quantifies the upper bound on avoidable coordination cost under a regime that coordinates all tasks uniformly.

We apply this framework to two complementary corpora. We classify 65 enterprise workflows across all 13 categories of the APQC Process Classification Framework [10] and find that 74% are monotonic under our decompositions. We extend this classification to 13,417 occupational tasks from the O*NET 29.1 database [11] spanning 22 SOC major groups using a calibrated LLM annotator [12, 13], and classify 42% as monotonic under multi-agent decomposition. These rates imply that up to 24–57% of coordination spending is unnecessary for correctness under uniform coordination. Multi-agent AI simulations then illustrate the distinction experimentally across three model families, where coordinated and uncoordinated conditions differ in whether agents can revise or reconcile one another's outputs and validity is assessed against pre-specified criteria.

Taken together, these results connect several lines of organizational theory. The Bridge Theorem contributes a formal criterion for a question that Thompson's [3] verbal classification left implicit and that Malone and Crowston's [14] program of characterizing dependencies left open, namely when is coordination necessary for correctness at all. The contribution is complementary to contemporary organization design work, which emphasizes that actual coordination requirements also depend on who knows what, who influences whom, and which coordination mechanism is deployed [4, 5]. Task topology can inform coordination intensity for correctness-critical work, whether in human organizations or AI systems.

Theory

Theoretical Bridge Between Interdependence and Monotonicity

Coordination is required for correctness when local actions can invalidate previously admissible outcomes. Distributed systems theory formalizes this condition using specification monotonicity: a task is monotonic if extending the execution history cannot invalidate previously admissible conclusions.

Organizational theory classifies tasks by patterns of interdependence instead. We show that these two traditions align except for the reciprocal case, where alignment is conditional on feedback type.

Establishing the correspondence requires three definitions. We define correctness as the property that a system's outputs satisfy a specified constraint under all admissible execution orders. We define coordination as any mechanism that enforces ordering or synchronization to prevent inconsistent intermediate states. A task requires coordination if correctness cannot be guaranteed under asynchronous, unordered execution. Formal definitions appear in SI Text S1.

Thompson distinguishes pooled, sequential, and reciprocal interdependence. Pooled tasks aggregate independent contributions, and their outputs combine via a join-semilattice [15]. Sequential tasks require ordered handoffs along Lamport's happened-before order [16], where each agent's computation is non-retractive. Reciprocal tasks involve mutual adjustment among agents, but reciprocal structure alone does not settle monotonicity. The relevant distinction is whether feedback merely adds information or whether it retracts or constrains previously admissible outcomes.

CALM provides the complementary criterion from distributed systems: a specification admits coordination-free execution if and only if it is monotonic [6, 7, 9]. Monotonicity is orthogonal to recursion or feedback as such. Recursive or feedback-rich tasks may still be monotonic if each history extension only enlarges what can be concluded. By contrast, specifications that require retraction, negation, or other shrinking of admissible outcomes require coordination. The following theorem states the bridge we establish.

Theorem 1 (Thompson-CALM Bridge)**.** *Under deterministic task evaluation, reliable message delivery, and eventual consistency in state propagation, let T be a multi-agent task. Then:*

(a) Pooled interdependence $\implies$ the task specification is monotone (coordination-free).

(b) Sequential interdependence $\implies$ the task specification is monotone under causal ordering (coordination-free with causal delivery).

Both cases hinge on non-retractive composition. For (a), sub-task monotonicity and the join-semilattice property guarantee that aggregate output only grows. For (b), the composition of non-retractive functions is non-retractive, so causal delivery suffices. Reciprocal interdependence alone does not determine monotonicity, but feedback that retracts or constrains previously admissible outcomes makes the specification non-monotone by definition and therefore requires coordination. Full proofs appear in SI Text S2.

The Bridge Theorem classifies pure sequential tasks as monotonic under non-retractive causal composition, but many real workflows contain feedback loops such as code review, quality inspection, and editorial passes. The Feedback Boundary resolves whether feedback pushes a task across the monotonicity boundary.

Proposition 1 (Feedback Boundary)**.** *Let T be a sequential task with feedback from a downstream agent to an upstream agent.*

(a) If the feedback is additive, so the upstream agent incorporates new information without retracting prior output, the specification remains monotone.

(b) If the feedback is retractive, so the upstream agent revises or replaces prior output, the specification is non-monotone.

Additive feedback preserves the non-retractive property established in Theorem 1(b), whereas retractive feedback violates it. This distinction maps onto Argyris and Schön's [17] boundary between single-loop refinement and double-loop revision.

The Coordination Tax

The Bridge Theorem identifies *which* tasks need coordination under the paper's formal assumptions. The Coordination Tax quantifies *how much* coordination cost is avoidable when that classification is ignored. Let $\mathcal{P}$ be a portfolio of n tasks with non-monotonic fraction $f \in [0, 1]$ and coordination cost multiplier $c > 1$. If every task receives uniform coordination, the unnecessary fraction of total spending is

$$T(f, c) = \frac{(1 - f)(c - 1)}{c} \tag{1}$$

Equation (1) is an upper bound on avoidable coordination cost under a regime in which all tasks receive uniform coordination regardless of structural necessity. It is not a direct measure of realized waste in organizations, which may already use differentiated coordination mechanisms such as standardization, modularization, routines, authority, and mutual adjustment.

The derivation is straightforward. Uniform cost nc minus selective cost $n(1-f+fc)$ equals $n(1-f)(c-1)$; dividing by nc gives T. As $f \to 0$ (no tasks need coordination), $T \to (c-1)/c$; as $f \to 1$ (all tasks need coordination), $T \to 0$.

Results

We classify two task corpora under the Bridge Theorem and illustrate the decision rule in multi-agent AI simulations.

Most Enterprise Workflows Are Monotonic (74%)

Our first corpus draws from the APQC Process Classification Framework [10], the most widely used enterprise process taxonomy (maintained since 1992). We classify 65 workflows at Levels 2–3 (team-level workflows with 2–5 agents), enumerating all eligible processes as an exhaustive census across all 13 APQC categories (see Materials and Methods).

Applying the Bridge Theorem to this corpus, we find that 48 of 65 tasks (73.8%) are monotonic (Table 1), of which 39 are fully coordination-free and 9 require only causal ordering. The remaining 17 (26.2%) require full coordination. These labels reflect the task decompositions and heuristic decision rules stated in Materials and Methods, with borderline cases defaulting to non-monotonic.

Breaking the result down by APQC category, we find systematic variation (Table 1). Risk & Compliance ranks highest at 100% because its tasks consist primarily of independent analyses and reports that merge without conflict, five categories share 80%, and Financial Resources ranks lowest at 57% because most of its tasks involve allocating shared budgets or credit across competing claims.

To understand what drives the 26% that require coordination under this framework, we examine the non-monotonic tasks individually and find that all 17 involve allocation of shared finite resources (budget, headcount, capacity, inventory). The pattern echoes Thompson's observation that reciprocal interdependence clusters at critical integration points [3]. Shared finite resources introduce negation or retractive constraint, which breaks monotonicity. Under these decompositions, the bulk of enterprise workflows can execute correctly without coordination.

Because these labels may depend on decomposition choices, we also performed a sensitivity analysis on a small sample of ten APQC tasks (five M/M-O and five NM, spread across APQC categories) reclassified under alternative reasonable assumptions about timing, shared-resource scope, or review semantics (SI Text S9). Seven of ten retained their monotonicity status, and three crossed the monotonicity boundary under the alternate decomposition. These boundary crossings show that

Table 1: **Most enterprise tasks (APQC) do not require coordination for correctness.** CALM monotonicity classification of 65 enterprise tasks across all 13 APQC categories. M: coordination-free; M-O: coordination-free under causal ordering; NM: coordination required.

APQC Category	M	M-O	NM	% Monotonic
Risk & Compliance	3	1	0	100%
Vision & Strategy	3	1	1	80%
Products & Services	3	1	1	80%
Marketing & Sales	3	1	1	80%
Service Delivery	3	1	1	80%
Customer Service	3	1	1	80%
Assets	3	0	1	75%
External Relations	3	0	1	75%
Business Capabilities	3	0	1	75%
Human Capital	4	0	2	67%
Information Technology	3	1	2	67%
Physical Products (info)	3	0	2	60%
Financial Resources	2	2	3	57%
Total	**39**	**9**	**17**	**73.8%**

decomposition choices can sometimes introduce or remove non-monotonicity by changing where coordination constraints are represented. Of the seven stable tasks, two also shifted from ordering-only to fully monotone, which does not affect the monotonicity rate. Taken together, most tasks retain the same monotonicity status under reasonable decomposition variation, and the 74% monotonicity rate remains decomposition-dependent rather than an invariant property of the tasks.

Nearly Half of Occupational Tasks Are Monotonic (42%)

To test whether monotonicity prevalence extends beyond enterprise workflows, we apply a calibrated LLM classifier ($\kappa \geq 0.88$ against author labels on APQC; see Materials and Methods) to 13,417 core task statements from the O*NET 29.1 occupational database [11] spanning 22 SOC major groups (Figure 1). Of these, 5,564 (41.5%, 95% CI [40.6, 42.3]) are monotonic. The rate is lower than APQC's 74%, which reflects the difference between team-level workflows (APQC) and individual work activities (O*NET). APQC processes decompose naturally into multi-agent sub-tasks, whereas O*NET tasks describe what a single worker does. The O*NET classification therefore characterizes the inferred coordination structure of each task under multi-agent decomposition.

Figure 1 shows the breakdown by SOC group. Monotonic tasks appear in every group, from Life and Physical Science (52%) to Management (32%). Groups dominated by independent observation and analysis (sciences, sales, healthcare support) rank highest, while Management ranks lowest because it concentrates resource allocation, staffing, and cross-unit negotiation. Full per-group results with 95% Wilson confidence intervals appear in SI Table S4. Under this decomposition, the scope for coordination avoidance extends beyond enterprise workflows to the broader occupational landscape.

Estimating the Coordination Tax

With both corpora classified, we can compute the Coordination Tax using Equation 1, $T(f, c) = (1 - f)(c - 1)/c$, which assumes that all tasks receive uniform coordination regardless of struc-

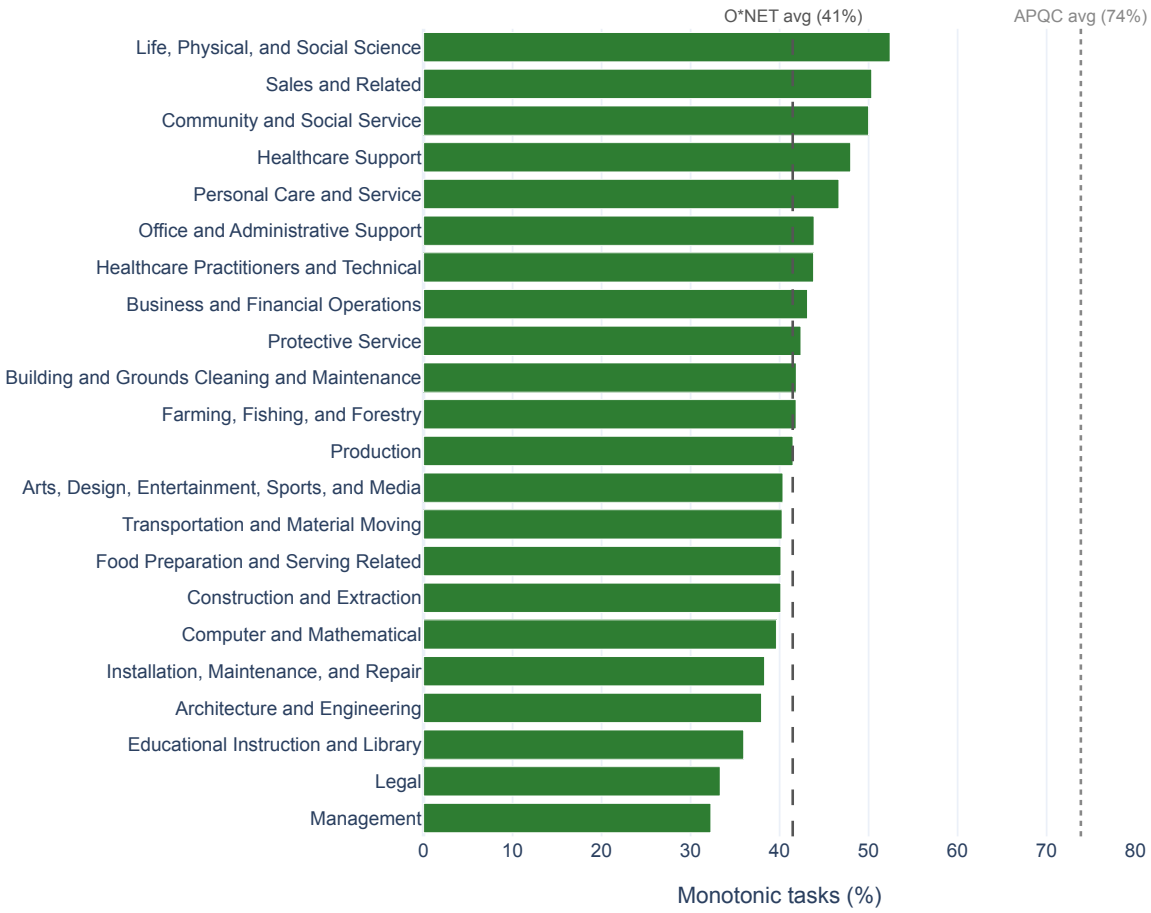

Figure 1: Monotonic tasks appear in every occupational category. Bars show monotonicity prevalence across 22 SOC major groups (13,417 O*NET tasks); every group exceeds 30%. Dashed line marks the O*NET average (41%); dotted line marks the APQC enterprise average (74%).

tural necessity. The non-monotonic fraction is $f = 17/65 \approx 0.26$ for APQC (Table 1) and $f = 7{,}852/13{,}416 \approx 0.58$ for O*NET (SI Table S4). Substituting these fractions and the simulation-measured overhead ratios ($c = 2.3$–$4.4\times$, SI Table S7) gives an APQC-derived tax of $T \approx 42$–57% and an O*NET-derived tax of $T \approx 24$–32%.

The gap between corpora reflects the unit of analysis: APQC describes team-level workflows where coordination cost concentrates, while O*NET describes individual work activities. In multi-agent AI systems specifically, published overhead estimates are higher ($c = 4$–$10\times$ [18, 19]), pushing the APQC range to $T = 56$–67%. Even at the most conservative estimate (O*NET, $c = 2.3\times$), the formula places nearly a quarter of coordination cost in the avoidable range. These quantities are upper bounds under stated assumptions, not direct estimates of realized organizational inefficiency, and organizations that already differentiate coordination across tasks may realize only a fraction.

Simulations Illustrate the Monotonicity Boundary

To illustrate the framework's prediction in a multi-agent AI setting, we run simulations where coordination can be cleanly toggled and costs are directly measurable in tokens (see Materials and Methods). In the coordinated condition, an orchestrator plans assignments, passes intermediate outputs between agents, and reconciles conflicts where needed. In the uncoordinated condition, agents receive only their sub-task descriptions and their outputs are concatenated without review. We select ten tasks spanning the monotonicity spectrum and run each under both conditions across three models (GPT-4.1 mini, Claude Haiku 4.5, Claude Sonnet 4.5; 10 repetitions at temperature 0). Validity is assessed against pre-specified task criteria by an LLM judge, with programmatic checks for the resource-constrained tasks.

Table 2: **Uncoordinated validity rate in simulations across three models.** Coordinated condition substantially recovers non-monotonic validity across models; full per-model rates appear in SI Text S8. Bold = 0%. M = monotonic, M-O = monotonic with ordering, NM = non-monotonic.

Task	Type	GPT-4.1-mini	Haiku 4.5	Sonnet 4.5
Strategy pillars	M	100%	100%	100%
Feature specs	M	100%	100%	100%
Marketing content	M	100%	100%	90%
Security audit	M	100%	100%	100%
Stage-gate	M-O	100%	100%	100%
Ticket escalation	M-O	100%	60%	70%
Budget alloc.	NM	**0%**	**0%**	**0%**
Backlog sprint	NM	**0%**	**0%**	**0%**
Production sched.	NM	**0%**	**0%**	**0%**
Headcount alloc.	NM	**0%**	**0%**	**0%**

The prediction holds across all three model families (Table 2). For GPT-4.1 mini, all six monotonic-or-ordered tasks achieve 100% validity in both conditions, while every non-monotonic task drops to 0% validity without coordination. The Claude models replicate the boundary result: uncoordinated non-monotonic tasks fail at 0% across all four tasks, while monotonic tasks remain near ceiling. Remaining variance on ordering-only tasks tracks model capability rather than violating the framework. Validity is binary in these runs, whereas organizational failures more often manifest as gradual degradation or rework. Coordinated runs consume 2.3–4.4 times more tokens (SI Table S7) and substantially recover non-monotonic validity.

Discussion

We show that for a substantial fraction of tasks, coordination is not required for correctness. The Bridge Theorem identifies which tasks those are through a partial correspondence between Thompson's taxonomy and CALM's monotonicity criterion, and the Coordination Tax bounds the avoidable coordination cost under a regime that coordinates all tasks uniformly. Across two corpora, 42–74% of tasks are monotonic under our decompositions, and 24–57% of coordination cost would be avoidable under uniform-coordination assumptions. A sensitivity analysis on APQC tasks shows that plausible alternative decompositions can move some tasks across the monotonicity boundary, especially when review is revisionary or shared resources are pre-partitioned, but most tasks retained their monotonicity status. Simulations illustrate the distinction operationally in a controlled

multi-agent AI setting.

The Bridge Theorem clarifies one dimension of interdependence that Thompson [3] highlighted but did not formalize, namely whether extending the task history can invalidate previously admissible outcomes. Under this criterion, pooled and sequential structures align naturally with monotone specifications, whereas reciprocal structures require coordination for correctness when feedback is retractive or constraining. This partial correspondence shows that reciprocal feedback need not be non-monotone.

The monotone form of reciprocal feedback is additive feedback, which can support coordination-free feedback loops even though retractive or constraining feedback remains non-monotone. Recognizing additive feedback as monotone expands the space of coordination-free specifications and points to a broader question for organization science. One possible reason this case has received less attention is that many organizational feedback processes are studied in settings where feedback revises, approves, constrains, or reallocates prior work. Future work can ask when monotone feedback is practically achievable, and whether recognizing it can inform organizational and AI workflows that accumulate information without invalidating prior outputs.

Specification monotonicity also responds to long-standing critiques of Thompson. Victor and Blackburn [20] argued that Thompson's typology mixes structural, behavioral, and evaluative dimensions of interdependence, and McCann and Ferry [21] called for operational measures beyond categorical labels. Specification monotonicity is one such semantic property, and it cuts across Thompson's categories. A reciprocal task with additive feedback can be monotonic, whereas a sequential task with retractive revision can be non-monotonic. Because the criterion is operationally testable by asking whether history extension can invalidate previously admissible conclusions, it speaks directly to both critiques.

This contribution is complementary to contemporary organization science. Task interdependence is distinct from agent interdependence, which concerns how agents' actions are coupled through learning, influence, or relational contracts, and from epistemic interdependence, which captures the distribution of knowledge across agents independent of task structure [4]. A microstructural perspective further emphasizes that coordination needs depend on authority, influence, and the fit between interdependence structures and coordination mechanisms [5]. Monotonicity is a property of the task specification alone. It is one input to coordination need alongside epistemic, agent-level, incentive, and mechanism-fit considerations. For organization scholars, the contribution is a testable semantic criterion that cuts across existing typologies rather than a new theory of organizations.

The Bridge Theorem also advances the research program that Malone and Crowston [14] outlined. They defined coordination as "managing dependencies among activities" and proposed that progress would come from characterizing dependency types and the coordination processes needed to manage them. The Bridge Theorem addresses a prior question: given a dependency type and a task specification, is coordination necessary at all for correctness? In that sense, it supplies a formal criterion that their framework did not provide.

The Coordination Tax isolates a category of coordination cost that is avoidable for correctness under a uniform-coordination regime. It does not imply that observed coordination in organizations is wasteful by that amount, because organizations often deploy differentiated mechanisms such as standardization, routines, authority, or modularization. The formula instead identifies the gap between a world that coordinates everything uniformly and a world that coordinates only when required for correctness. Baldwin and Clark's [22] account of modularity fits naturally here. Modularity can reduce the non-monotonic share of a task portfolio by localizing retractive dependencies within a thinner integration layer.

These findings carry practical implications for multi-agent systems, human or artificial. Pooled tasks can often use fire-and-forget execution with merge by join. Sequential tasks can often rely

on causal delivery without full mutual adjustment. Only the non-monotonic subset requires conflict resolution, synchronization, or other stronger coordination mechanisms. For AI systems in particular, this suggests that agent topology should mirror task dependency structure rather than default managerial hierarchy. For multi-agent system designers, this is also an invitation to draw on organization-design scholarship rather than around it, treating monotonicity as one input into a broader design problem.

The framework and evidence have several limitations. First, CALM guarantees correctness, not broader performance objectives, so independently written sections may be valid yet stylistically inconsistent, slow, redundant, or poorly prioritized. Second, misclassification risk is asymmetric: a false monotonic label endangers correctness, whereas a false non-monotonic label mainly adds unnecessary coordination cost. Third, the monotonicity labels are decomposition-dependent rather than invariant properties of the tasks, and alternative assumptions about timing, shared-resource scope, or review semantics can move some tasks across the boundary. Fourth, the simulations use binary validity outcomes in a controlled AI-agent sandbox rather than field observations of organizations, whereas real organizational failures often appear as delays, rework, partial inconsistency, or degraded quality rather than clean invalidity. Finally, LLM-assisted classification, as deployed here, is a calibrated screening tool rather than a substitute for verification, and the resulting labels may vary across prompts and model implementations.

A few directions would extend this work. First, a neurosymbolic approach would give stronger certainty about any particular task by coupling an LLM proposer with a formal verifier and enforcing the monotone protocol through implementation. Second, field observations of organizations would test how the monotonicity boundary manifests in realized workflow behavior, complementing the controlled AI-agent simulations reported here. Third, epistemic interdependence could be connected more directly to distributed-systems accounts of common knowledge, including the impossibility result of Halpern and Moses and recent complexity extensions to CALM [23, 24]. Together, these would sharpen the framework from a theoretical upper bound under stated assumptions into directly testable claims about per-task correctness and observed coordination.

The broader implication is that whether a task requires coordination for correctness can be treated as a computable property of a formal task representation, providing a principled starting point for allocating coordination intensity. The strongest claim supported by the present evidence is that the framework identifies a theoretical upper bound on avoidable coordination under specified assumptions, not that the reported percentages represent realized inefficiency in any particular organization. As organizations and AI systems grow more complex, distinguishing correctness-critical coordination from coordination adopted by convention may become as important as improving coordination itself.

Materials and Methods

APQC corpus construction. We draw our primary task corpus from version 7.4 of the APQC Process Classification Framework (Cross Industry) [10], the most widely used enterprise process taxonomy. We define tasks at Levels 2–3 (2–5 agents, team-level workflows). A process qualifies if it decomposes into 2+ agent sub-tasks, involves information processing, produces a defined output, and spans industries. We enumerate all eligible processes as an exhaustive census, yielding 65 tasks across all 13 APQC categories (SI Table S3). For each task, we classify interdependence structure using Thompson's taxonomy, apply the Bridge Theorem to predict CALM classification, and verify against five heuristic tests (SI Text S3). Borderline cases default to non-monotonic. These labels are best understood as structured proxy judgments about monotonicity, not direct formal verifica-

tion of every workflow. We are careful to distinguish the theoretical construct (*i.e.*, specification monotonicity under history growth) from the empirical proxy (*i.e.*, structured judgments under a stated decomposition).

O*NET classification procedure. To scale beyond the 65 APQC tasks, we calibrate an LLM classifier against the author-classified APQC corpus. We use GPT-4.1 mini and Claude Sonnet 4.5, each under two prompting conditions (3 runs at temperature 0), achieving $\kappa \geq 0.88$ on the binary monotonic-versus-non-monotonic distinction (SI Table S5). Every misclassification labels a monotonic task as non-monotonic. We apply the calibrated classifier (GPT-4.1 mini, heuristic condition) to 13,417 core task statements from the O*NET 29.1 occupational database [11] spanning 22 SOC major groups, following Eloundou et al. [12], who used GPT-4 to annotate O*NET tasks for LLM exposure. Their theoretical exposure measures have since been empirically validated against observed LLM usage (97% of observed Claude-usage tasks fall in categories Eloundou rated as theoretically feasible [13]). The resulting labels are inferred under the paper's decomposition assumptions and should not be read as direct measures of occupational coordination in practice.

Simulation design. Ten tasks span the monotonicity spectrum (four monotonic, two requiring ordering only, four non-monotonic) under two conditions. In the coordinated condition, an orchestrator LLM plans assignments, passes intermediate outputs, reviews for consistency, and reconciles conflicts for the non-monotonic tasks. In the uncoordinated condition, agents receive only their sub-task descriptions and outputs are concatenated mechanically, with no review or reconciliation layer. An LLM-as-judge evaluator reasons step-by-step against pre-specified validity criteria before issuing a VALID or INVALID verdict. For the resource-constrained tasks, we supplement the judge with programmatic checks as reported in SI Text S8. We interpret these simulations as controlled illustrations in an AI setting rather than as direct evidence about organizational field behavior. We run 10 repetitions at temperature 0 across three models (GPT-4.1 mini, Claude Haiku 4.5, Claude Sonnet 4.5). Full task specifications, validity criteria, and token cost ratios appear in SI Text S8.

Data availability. The APQC Process Classification Framework is available at apqc.org. The O*NET 29.1 database is publicly available at onetonline.org. All classification data and simulation code will be deposited at `https://github.com/harangju/coordination-avoidability` upon publication.

Author contributions. H.J. designed the study, developed the theory, performed the analysis, and wrote the paper.

Competing interests. The author declares no competing interests.

References

[1] Mohamad Salim, Jasmine Latendresse, SayedHassan Khatoonabadi, and Emad Shihab. Tokenomics: Quantifying where tokens are used in agentic software engineering. *arXiv preprint arXiv:2601.14470*, 2026.

[2] 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. Towards a science of scaling agent systems. *arXiv preprint arXiv:2512.08296*, 2025.

[3] James D. Thompson. *Organizations in Action: Social Science Bases of Administrative Theory*. McGraw-Hill, New York, 1967.

[4] Phanish Puranam, Marlo Raveendran, and Thorbjørn Knudsen. Organization design: The epistemic interdependence perspective. *Academy of Management Review*, 37(3):419–440, 2012.

[5] Phanish Puranam. *The Microstructure of Organizations*. Oxford University Press, Oxford, 2018.

[6] Joseph M. Hellerstein and Peter Alvaro. Keeping CALM: When distributed consistency is easy. *Communications of the ACM*, 63(9):72–81, 2020.

[7] Tom J. Ameloot, Frank Neven, and Jan Van den Bussche. Relational transducers for declarative networking. *Journal of the ACM*, 60(2):15:1–15:38, 2013.

[8] Peter Bailis, Alan Fekete, Michael J. Franklin, Ali Ghodsi, Joseph M. Hellerstein, and Ion Stoica. Coordination avoidance in database systems. *Proceedings of the VLDB Endowment*, 8(3):185–196, 2015.

[9] Joseph M. Hellerstein. The coordination criterion. *arXiv preprint arXiv:2602.09435*, 2026.

[10] APQC. APQC process classification framework (PCF) — cross industry, version 7.4. `https://www.apqc.org/process-frameworks`, 2024. Industry-standard enterprise process taxonomy since 1992.

[11] National Center for O*NET Development. O*NET 29.1 database, 2024. Sponsored by the U.S. Department of Labor.

[12] Tyna Eloundou, Sam Manning, Pamela Mishkin, and Daniel Rock. GPTs are GPTs: Labor market impact potential of LLMs. *Science*, 384(6702):1306–1308, 2024.

[13] Maxim Massenkoff and Peter McCrory. Labor market impacts of AI: A new measure and early evidence, 2026. Anthropic Economic Index report.

[14] Thomas W. Malone and Kevin Crowston. The interdisciplinary study of coordination. *ACM Computing Surveys*, 26(1):87–119, 1994.

[15] Marc Shapiro, Nuno Preguiça, Carlos Baquero, and Marek Zawirski. Conflict-free replicated data types. In *Stabilization, Safety, and Security of Distributed Systems (SSS)*, pages 386–400. Springer, 2011.

[16] Leslie Lamport. Time, clocks, and the ordering of events in a distributed system. *Communications of the ACM*, 21(7):558–565, 1978.

[17] Chris Argyris and Donald A. Schön. *Organizational Learning: A Theory of Action Perspective*. Addison-Wesley, Reading, MA, 1978.

[18] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W. White, Doug Burger, and Chi Wang. AutoGen: Enabling next-gen LLM applications via multi-agent conversation. In *Conference on Language Modeling (COLM)*, 2024.

[19] Sirui Hong, Mingchen Zhuge, Jiaqi Chen, Xiawu Zheng, Yuheng Cheng, Ceyao Zhang, Jinlin Wang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. MetaGPT: Meta programming for a multi-agent collaborative framework. In *International Conference on Learning Representations (ICLR)*, 2024.

[20] Bart Victor and Richard S. Blackburn. Interdependence: An alternative conceptualization. *Academy of Management Review*, 12(3):486–498, 1987.

[21] Joseph E. McCann and Diane L. Ferry. An approach for assessing and managing inter-unit interdependence. *Academy of Management Review*, 4(1):113–119, 1979.

[22] Carliss Y. Baldwin and Kim B. Clark. *Design Rules, Volume 1: The Power of Modularity*. MIT Press, Cambridge, MA, 2000.

[23] Joseph Y. Halpern and Yoram Moses. Knowledge and common knowledge in a distributed environment. *Journal of the ACM*, 37(3):549–587, 1990.

[24] Joseph M. Hellerstein. On the complexity of determinations. *arXiv preprint arXiv:2603.28031*, 2026.

Supporting Information for When Coordination Is Avoidable: A Monotonicity Analysis of Organizational Tasks

Harang Ju
Carey Business School, Johns Hopkins University

Contents

1 Formal Preliminaries

This section states the formal definitions underlying the execution model and Thompson's interdependence types in two steps. We first restate the CALM execution model in multi-agent terms, replacing distributed nodes with agents. We then formalize Thompson's three interdependence categories in that same apparatus.

CALM in Multi-Agent Terms

The following definitions recast the CALM framework [1, 2] using agents rather than distributed nodes.

Definition S1 (Multi-agent task execution)**.** *A multi-agent task execution is a tuple* $(\mathcal{A}, E, \to, \text{Obs})$ *where* $\mathcal{A} = \{a_1, \ldots, a_n\}$ *is a finite set of agents;* E *is a set of events;* $\to$ *is the happened-before partial order on* E *[3]; and* $\text{Obs} : \mathcal{H} \to \mathcal{P}(\mathcal{O})$ *maps each history to a set of admissible observable outcomes, where* $\mathcal{H}$ *is the set of all histories of* $(E, \to)$ *and* $\mathcal{O}$ *is the set of all possible outcomes.*

A *history* H is a downward-closed subset of $(E, \to)$: if $e \in H$ and $e' \to e$, then $e' \in H$.

Definition S2 (History extension)**.** H_1 *is a prefix of* H_2*, written* $H_1 \sqsubseteq H_2$*, if* $H_1 \subseteq H_2$ *and* H_1 *is downward-closed in* $(H_2, \to)$*.*

Definition S3 (Outcome order)**.** *An outcome order* $\preceq$ *is a partial order on* $\mathcal{O}$ *that captures when one outcome refines another. The order is task-specific. For tasks whose outputs are sets of results, the natural order is set inclusion:* $o_1 \preceq o_2$ *iff* $o_1 \subseteq o_2$*. For tasks whose outputs are tuples (e.g., resource allocations), the natural order is component-wise:* $(x_1, \ldots, x_n) \preceq (y_1, \ldots, y_n)$ *iff* $x_i \leq y_i$ *for all* i*.*

Definition S4 (Coordination-free implementation)**.** *An implementation is* coordination-free *if (i) it only exposes correct outcomes and (ii) it permits all causally admissible histories: no agent is ever blocked to prevent future inconsistency [1, 2].*

Definition S5 (Monotone and non-monotone specifications)**.** *A task specification (the pair* $(\text{Obs}, \preceq)$ *from a given execution) is* monotone *if for all histories* $H_1 \sqsubseteq H_2$ *and all outcomes* $o \in \text{Obs}(H_1)$*, there exists* $o' \in \text{Obs}(H_2)$ *with* $o \preceq o'$*. It is* non-monotone *if there exist* $H_1 \sqsubseteq H_2$ *and* $o \in \text{Obs}(H_1)$ *such that no* $o' \in \text{Obs}(H_2)$ *satisfies* $o \preceq o'$*.*

By the CALM Coordination Theorem, a task specification admits a coordination-free implementation if and only if it is monotone [1, 2, 4].

Thompson's Interdependence (Formalized)

Thompson's three categories each impose distinct structural constraints on the agent dependency graph and the task's output space. A functional-programming analogy sharpens the intuition. Pooled tasks compose like pure functions: independent, freely parallelizable, and mergeable by accumulation. Sequential tasks thread output through a monotone pipeline under causal order. Reciprocal tasks introduce feedback among agents, but feedback alone does not determine monotonicity. The key semantic distinction is whether extending the history only adds admissible information or instead retracts or constrains previously admissible outcomes. Mutable shared state is neither necessary nor sufficient for non-monotonicity.

Definition S6 (Pooled interdependence)**.** *A task has* pooled interdependence *if: (1) it decomposes into sub-tasks $T_1, \ldots, T_n$ assignable to agents $a_1, \ldots, a_n$; (2) all cross-agent event pairs are concurrent (no causal dependencies); (3) each sub-task specification is monotone; and (4) the output space $(\mathcal{O}, \preceq)$ is a join-semilattice, and the aggregate outcome is the join $\mu(o_1, \ldots, o_n) = o_1 \sqcup \cdots \sqcup o_n$. This is the algebraic structure underlying CRDTs [5].*

Definition S7 (Sequential interdependence)**.** *A task has* sequential interdependence *if: (1) for some permutation π of $\{1, \ldots, n\}$, agents are ordered $a_{\pi(1)}, \ldots, a_{\pi(n)}$ with causal dependencies from $a_{\pi(k)}$ to $a_{\pi(k+1)}$; (2) the dependency graph is acyclic; and (3) each agent $a_{\pi(k)}$ has a local output function $f_k : \mathcal{O} \to \mathcal{O}$ that is non-retractive (order-preserving): $o \preceq o' \implies f_k(o) \preceq f_k(o')$.*

Definition S8 (Reciprocal interdependence)**.** *A task has* reciprocal interdependence *if: (1) the agent dependency graph contains a cycle; and (2) at least one feedback path can change the set of admissible outcomes for another agent. Reciprocal tasks need not all be non-monotone. They become non-monotone when the feedback is retractive or otherwise constraining, so that some previously admissible outcomes are no longer admissible after history extension.*

2 Proofs

Bridge Theorem Proofs

The following proofs verify that the structural conditions in Definitions S6 and S7 compose to yield the claimed monotonicity properties.

Proof of Theorem 1(a): Pooled $\implies$ Monotone. Let $H \sqsubseteq H'$ be histories. Decompose by agent: $H = H^1 \cup \cdots \cup H^n$, and likewise $H' = (H')^1 \cup \cdots \cup (H')^n$. Since cross-agent events are concurrent (condition 2), $H^i \sqsubseteq (H')^i$ for each agent i. Sub-task monotonicity (condition 3) ensures each agent's output refines under history extension. The join-semilattice property (condition 4) then guarantees $o_1 \sqcup \cdots \sqcup o_n \preceq o'_1 \sqcup \cdots \sqcup o'_n$, so the aggregate is monotone. □

Proof of Theorem 1(b): Sequential $\implies$ Monotone under Causal Ordering. The composition of non-retractive functions is non-retractive: if $o \preceq o'$, then $f_1(o) \preceq f_1(o')$, hence $f_2(f_1(o)) \preceq f_2(f_1(o'))$, and so on. Causal delivery ensures each pipeline stage processes refined input. Therefore, for every pair of causally ordered histories $H_1 \sqsubseteq H_2$, every admissible outcome at H_1 has a refinement at H_2. The specification is monotone over causally ordered histories. □

Remark on reciprocal feedback. Reciprocal interdependence alone does not determine monotonicity. When reciprocal feedback is retractive or constraining, some action by a_j can remove outcomes previously admissible for a_i. Let H_1 be a history in which a_i has acted but the relevant constraining action has not yet occurred, and let $H_2 = H_1 \cup \{e_j\}$ include that action. If the extension removes a previously admissible outcome $o \in \mathrm{Obs}(H_1)$ and no $o' \in \mathrm{Obs}(H_2)$ satisfies $o \preceq o'$, the specification is non-monotone by Definition S5. Shared-budget tasks illustrate the case: if agents share budget $B = 100$ with invariant $\mathrm{spend}_1 + \mathrm{spend}_2 \leq B$, one agent's later request can make an earlier allocation inadmissible. Additive reciprocal feedback need not have this property.

Feedback Boundary Proofs

Proof of Proposition 1(a). Let $H_1 \sqsubseteq H_2$ where H_2 includes feedback message m from a_j to a_i. Since the feedback is additive, the upstream agent incorporates m without retracting prior output, so its updated output satisfies $o_i \preceq o'_i$. Since downstream functions are non-retractive, the refinement propagates through the remainder of the pipeline. Thus every admissible outcome at H_1 has a refinement at H_2, and the specification remains monotone. □

Proof of Proposition 1(b). Retractive feedback yields an updated upstream output o_i'' such that a previously admissible outcome is no longer refinable, that is, $o_i \not\preceq o_i''$ for some prior output o_i. Hence there exists $o \in \mathrm{Obs}(H_1)$ with no refinement in $\mathrm{Obs}(H_2)$, violating monotonicity. Because the retractive feedback closes a causal cycle and shrinks the admissible set, the task falls under reciprocal interdependence as defined above. □

3 Heuristic Classification Tests

The Bridge Theorem requires classifying a task's interdependence structure, which in turn requires reasoning about dependency graphs and output spaces. For practitioners who need a quick classification without formal verification, Table S1 distills the theorem, feedback boundary, and definitions into five plain-language tests. Each test operationalizes a sufficient condition for non-monotonicity identified by the framework. Tests 1–4 screen for non-monotonicity; Test 5 confirms monotonicity. The tests are conservative by design, with errors falling toward over-coordination rather than under-coordination.

Table S1: **Five plain-language heuristic tests for monotonicity classification.** Tests 1–4 screen for non-monotonicity; Test 5 confirms monotonicity.

Step	Test	Question	If YES	If NO
1	Shared Finite Resource	Do sub-tasks draw from a shared pool?	Non-mono.	Continue
2	Negation	Do constraints require ensuring absence?	Non-mono.	Continue
3	Retraction	Can one agent's output make another's previously admissible output incorrect?	Non-mono.	Continue
4	Dependency	Does correctness depend on specific content of another's output?	Non-mono.	Continue
5	Merge	Can all outputs always be combined validly?	Monotonic	Non-mono.

Table S2: **Worked examples applying the heuristic tests to six representative tasks.**

#	Task	Triggering test	Monotonicity	Thompson
1	Market analysis across 5 regions	Merge: YES	Mono.	Pooled
2	Cross-functional hiring: 3 teams, 10 candidates	Shared Resource: YES	Non-mono.	Reciprocal
3	R&D budget: 4 divisions, $2M	Shared Resource: YES	Non-mono.	Reciprocal
4	Quarterly planning: depts. write objectives	Merge: YES	Mono.	Pooled
5	Multi-chapter style guide: consistent terminology	Dependency: YES	Non-mono.	Reciprocal
6	Sequential report with additive review notes	Merge plus causal order	Mono.	Sequential

4 Extended Related Work

Organizational Theory

Several traditions in organizational theory address when and why coordination is needed. We isolate a formal criterion for correctness-critical coordination under a specified task representation, then relate it to broader organization-design frameworks.

Organizations are information-processing systems, and the structure of a task determines how much processing capacity its execution requires. Galbraith [6] argued that greater task uncertainty demands greater organizational capacity, either by reducing the need for information (through slack resources and self-contained tasks) or by increasing it (through vertical information systems and lateral relations). The Bridge Theorem identifies one formal boundary within this framework. Monotonic tasks have lower correctness-critical information-processing requirements because new information does not invalidate prior work. Non-monotonic tasks require the additional capacity that coordination provides.

Puranam, Raveendran, and Knudsen [7] draw a sharper distinction between task interdependence and epistemic interdependence. Even when the task structure is fixed, coordination needs can vary with what different agents know about one another, the task, and the environment. That distinction matters here. Our framework characterizes when the task specification itself requires coordination for correctness. It does not characterize all the knowledge-based reasons organizations may still coordinate monotonic work. Our criterion therefore applies to task interdependence under a specified representation, and epistemic interdependence can create additional coordination needs that fall outside the scope of our result.

Thompson's typology has also attracted specific criticism for conflating dimensions of interdependence and for lacking operational measures. Victor and Blackburn [8] argued that Thompson's three categories mix structural, behavioral, and evaluative aspects, and McCann and Ferry [9] called for measurable inter-unit interdependence indices beyond typological labels. Specification monotonicity addresses these concerns from a new angle. It isolates one well-defined semantic property that cuts across Thompson's categories, and it is operationally testable via history extension rather than requiring categorical judgment. Our treatment is therefore consistent with Victor and Blackburn's call to disaggregate interdependence, and with McCann and Ferry's call for an operational criterion.

The monotonicity boundary also intersects with the fit between interdependence and coordination mechanisms. Mintzberg [10] catalogued five mechanisms ranging from mutual adjustment through direct supervision to three forms of standardization (of work processes, outputs, and skills), with organizations reserving the costliest for the most unpredictable work. Puranam's microstructural perspective likewise emphasizes the mapping between interdependence, authority, and influence structures [11]. Our argument is complementary. It identifies when stronger mechanisms are required for correctness at all. Choosing among feasible mechanisms remains a separate organization-design problem.

A related structural insight is that complex systems are nearly decomposable. Simon [12] argued that effective organizations consist of semi-independent modules with strong internal and weak external coupling. Monotonic tasks formalize one aspect of this weak coupling, because modules can operate independently when their outputs never conflict under history growth. Colfer and Baldwin [13] extend the modularity argument in their survey of the "mirroring hypothesis," finding that organizational structure mirrors technical architecture most strongly when interdependence is high. The Bridge Theorem helps explain the asymmetry. Constraining reciprocal tasks demand tighter alignment between organizational and technical boundaries, whereas pooled and sequential

monotone tasks allow looser coupling.

The framework also sharpens the economic analysis of coordination. Coase [14] argued that firms exist to reduce the transaction costs of market coordination, and Williamson [15] formalized this into transaction cost economics, where governance structures minimize these costs. Before asking how to minimize coordination costs, however, one can ask whether coordination is required for correctness at all. Under the paper's assumptions, coordination cost on monotonic tasks buys no correctness benefit. Ouchi [16] extended this reasoning to the choice among markets, bureaucracies, and clans, arguing that the right governance form depends on output measurability and goal alignment. Our framework narrows the scope of that choice by isolating the subset of tasks for which stronger coordination is correctness-critical.

Finally, the framework connects to group performance and AI. Steiner's [17] process loss model decomposes group performance as Actual = Potential − Process Losses. The Coordination Tax formalizes which of those losses are avoidable for correctness under a uniform-coordination assumption. More recently, Lee and Makridis [18] developed a model characterizing how AI affects the coordination demands of individual work activities. Their analysis complements ours in that we span both individual (O*NET) and team-level (APQC) tasks while keeping the contribution tightly scoped to correctness-critical coordination.

Malone-Crowston Dependency Mapping

The organizational frameworks above describe coordination qualitatively. A more direct formal link exists through the Malone-Crowston dependency taxonomy [19], which classifies inter-task dependencies into four types.

Malone-Crowston type	Thompson mapping	CALM classification	Coordination need
Task-subtask	Pooled	Monotone	None
Producer-consumer	Sequential	Monotone (causal)	Ordering only
Shared resources	Reciprocal	Non-monotone	Full coordination
Simultaneity	Often reciprocal	Often non-monotone	Often full coordination

This mapping should be read as a useful correspondence, not as an exhaustive equivalence. In particular, reciprocal structures with additive feedback can remain monotonic, and simultaneity can matter for reasons other than non-monotonicity depending on the specification.

I-Confluence A related formalization from the database literature captures the same intuition from the consistency perspective. Given system invariants $\mathcal{I}$ and agent operations $\{o_1, \ldots, o_n\}$, the operations are I-confluent with respect to $\mathcal{I}$ if, for any two agents executing independently from valid states, merging their states always satisfies $\mathcal{I}$ [2, 20]. Tasks with shared-resource or simultaneity dependencies fail I-confluence for the same reason they are non-monotone, because independent execution can produce states that violate global invariants.

5 Full APQC Corpus

Table S3 lists all 65 APQC tasks with their Thompson interdependence type and CALM monotonicity label. Tasks are grouped by APQC category and classified at Level-2/3 granularity (team-level workflows with 2–5 agents).

Table S3: **Full 65-task corpus with APQC categories, Thompson classifications, and CALM monotonicity labels.** Tasks are classified at APQC Level-2/3 granularity.

#	APQC Category	Task	Thompson	CALM
5	Vision & Strategy	Allocate strategic investment budget	Reciprocal	NM
3	Vision & Strategy	Assess organizational capabilities	Pooled	M
2	Vision & Strategy	Conduct competitive landscape analysis	Pooled	M
4	Vision & Strategy	Develop and cascade KPIs from strategy	Sequential	M-O
1	Vision & Strategy	Develop strategic plan components	Pooled	M
6	Products & Services	Conduct market requirements research	Pooled	M
7	Products & Services	Design product feature specifications	Pooled	M
8	Products & Services	Evaluate vendor proposals for components	Pooled	M
10	Products & Services	Prioritize product backlog with fixed capacity	Reciprocal	NM
9	Products & Services	Stage-gate product development review	Sequential	M-O
15	Marketing & Sales	Allocate marketing campaign budget	Reciprocal	NM
12	Marketing & Sales	Analyze regional sales performance	Pooled	M
13	Marketing & Sales	Conduct customer segmentation analysis	Pooled	M
11	Marketing & Sales	Develop multi-channel marketing content	Pooled	M
14	Marketing & Sales	Qualify and route inbound leads	Sequential	M-O
20	Physical Products (info)	Allocate warehouse inventory to orders	Reciprocal	NM
17	Physical Products (info)	Analyze supplier quality metrics	Pooled	M
16	Physical Products (info)	Develop demand forecasts by product line	Pooled	M
18	Physical Products (info)	Plan logistics routes for distribution	Pooled	M
19	Physical Products (info)	Schedule production across shared lines	Reciprocal	NM
22	Service Delivery	Analyze service quality metrics by region	Pooled	M
25	Service Delivery	Assign service technicians to work orders	Reciprocal	NM
23	Service Delivery	Compile service catalog entries	Pooled	M
21	Service Delivery	Document service delivery procedures	Pooled	M
24	Service Delivery	Process service request through fulfillment pipeline	Sequential	M-O
26	Customer Service	Analyze customer complaint categories	Pooled	M
30	Customer Service	Assign limited warranty replacement inventory	Reciprocal	NM
27	Customer Service	Draft FAQ and knowledge base articles	Pooled	M
28	Customer Service	Generate customer satisfaction survey reports	Pooled	M
29	Customer Service	Triage and route support tickets through escalation pipeline	Sequential	M-O
35	Human Capital	Allocate headcount across departments	Reciprocal	NM
32	Human Capital	Compile employee engagement survey analysis	Pooled	M
31	Human Capital	Develop job descriptions for open roles	Pooled	M
33	Human Capital	Develop training curriculum for compliance topics	Pooled	M
34	Human Capital	Review and benchmark compensation data	Pooled	M
36	Human Capital	Select candidates from shared applicant pool	Reciprocal	NM

#	APQC Category	Task	Thompson	CALM
41	Information Technology	Allocate cloud compute budget across teams	Reciprocal	NM
37	Information Technology	Conduct security vulnerability assessments	Pooled	M
39	Information Technology	Develop software test suites for independent modules	Pooled	M
38	Information Technology	Document IT system architecture	Pooled	M
40	Information Technology	Process change request through ITIL pipeline	Sequential	M-O
42	Information Technology	Schedule deployment windows on shared environments	Reciprocal	NM
48	Financial Resources	Allocate capital expenditure across projects	Reciprocal	NM
47	Financial Resources	Allocate operating budget across divisions	Reciprocal	NM
44	Financial Resources	Analyze expense reports by department	Pooled	M
49	Financial Resources	Assign credit limits from shared credit facility	Reciprocal	NM
46	Financial Resources	Perform tax compliance analysis by jurisdiction	Sequential	M-O
43	Financial Resources	Prepare financial statements by entity	Pooled	M
45	Financial Resources	Process accounts payable through approval pipeline	Sequential	M-O
53	Assets	Allocate shared vehicle fleet to departments	Reciprocal	NM
51	Assets	Analyze asset depreciation schedules	Pooled	M
50	Assets	Conduct asset condition assessments	Pooled	M
52	Assets	Inventory and tag enterprise assets by location	Pooled	M
55	Risk & Compliance	Audit regulatory compliance by domain	Pooled	M
54	Risk & Compliance	Conduct risk assessments across business functions	Pooled	M
56	Risk & Compliance	Develop business continuity plans by department	Pooled	M
57	Risk & Compliance	Process regulatory filing through review pipeline	Sequential	M-O
61	External Relations	Allocate corporate sponsorship budget	Reciprocal	NM
60	External Relations	Compile ESG disclosures by reporting dimension	Pooled	M
59	External Relations	Develop community engagement reports	Pooled	M
58	External Relations	Prepare government affairs briefings by jurisdiction	Pooled	M
65	Business Capabilities	Allocate improvement initiative budget	Reciprocal	NM
62	Business Capabilities	Benchmark operational metrics against industry peers	Pooled	M
63	Business Capabilities	Conduct process maturity assessments	Pooled	M
64	Business Capabilities	Evaluate knowledge management practices	Pooled	M

6 O*NET Full Results

Table S4 reports monotonicity prevalence by SOC major group with 95% Wilson confidence intervals.

Table S4: **O*NET monotonicity prevalence by SOC major group.** M = coordination-free, M-O = requires ordering only, NM = requires coordination. 95% Wilson CIs.

SOC Major Group	M	M-O	NM	% Monotonic	95% CI
Life, Physical, and Social Science	404	83	442	52%	[49, 56]
Sales and Related	131	14	143	50%	[45, 56]
Community and Social Service	99	9	108	50%	[43, 57]
Healthcare Support	111	20	142	48%	[42, 54]
Personal Care and Service	158	31	216	47%	[42, 52]
Office and Administrative Support	219	57	353	44%	[40, 48]
Healthcare Practitioners and Technical	504	95	767	44%	[41, 46]
Business and Financial Operations	248	38	377	43%	[39, 47]
Protective Service	166	7	235	42%	[38, 47]
Building and Grounds Cleaning and Maintenance	44	5	68	42%	[33, 51]
Farming, Fishing, and Forestry	46	8	75	42%	[34, 50]
Production	413	137	775	42%	[39, 44]
Arts, Design, Entertainment, Sports, and Media	178	34	313	40%	[36, 45]
Transportation and Material Moving	201	23	332	40%	[36, 44]
Food Preparation and Serving Related	94	12	158	40%	[34, 46]
Construction and Extraction	256	98	528	40%	[37, 43]
Computer and Mathematical	177	47	341	40%	[36, 44]
Installation, Maintenance, and Repair	216	79	475	38%	[35, 42]
Architecture and Engineering	273	79	575	38%	[35, 41]
Educational Instruction and Library	396	50	794	36%	[33, 39]
Legal	24	4	56	33%	[24, 44]
Management	243	33	579	32%	[29, 35]
Total	**4601**	**963**	**7852**	**41%**	**[41, 42]**

7 LLM Classifier Validation

The 65 APQC tasks were classified by author judgment. To assess whether LLMs can reproduce this proxy classification, and to support extending it to larger corpora where manual labeling is infeasible, we ran two LLMs (GPT-4.1 mini and Claude Sonnet 4.5) on the same 65 tasks under two prompting conditions, *heuristic* (the five-test decision table from Table S1) and *raw* (definitions only, no heuristics). The heuristic prompt provides the task description and the five-test decision table (Table S1); the raw prompt provides the task description and the definitions of monotone and non-monotone specifications without the heuristic tests. Full prompt text is available in the code repository. Each condition was run 3 times at temperature 0. Following Eloundou et al. [21], who used GPT-4 to annotate ∼19,000 O*NET task statements for LLM exposure, we treat LLM annotation as a scalable proxy for expert judgment when calibrated against a hand-labeled sample. Author labels serve as the calibration target, not as direct ground-truth proofs of monotonicity.

All four configurations achieve ≥95% accuracy and $\kappa \geq 0.88$ (Table S5). The error pattern is consistent across models. Every misclassification labels a monotonic task as non-monotonic (false

Table S5: **Classification accuracy of LLM-based monotonicity classifiers on the 65-task corpus.** Metrics computed on 2-class labels (M+M-O vs. NM) from Run 1. Self-consistency measured across 3 runs.

Model	Condition	Acc.	Mono F1	NM F1	Macro F1	κ	Consistency
GPT-4.1 mini	Heuristic	0.954	0.968	0.919	0.943	0.887	61/65
GPT-4.1 mini	Raw	0.985	0.989	0.971	0.980	0.961	64/65
Claude Sonnet 4.5	Heuristic	0.985	0.989	0.971	0.980	0.961	64/65
Claude Sonnet 4.5	Raw	0.954	0.968	0.919	0.943	0.887	62/65

non-monotonic), never the reverse. This bias is conservative because it over-coordinates rather than under-coordinates.

8 Simulation Protocol

Tasks Ten tasks from the corpus span the monotonicity spectrum (Table S6). We decompose each task into 3–5 specialist agents, fixing sub-task descriptions and validity criteria before execution.

Coordinated condition An orchestrator LLM (GPT-4.1 mini, temperature 0) manages the full execution pipeline. It first receives the task description and plans sub-task assignments. It then formats and delivers each assignment to the corresponding agent. Agents execute in order; for sequential tasks, the orchestrator passes prior-stage output to each downstream agent. After all agents complete, the orchestrator reviews the collected outputs for consistency. For non-monotonic tasks, it reconciles conflicting requests to produce a final allocation within the shared constraints. The coordinated token totals in Table S7 include all tokens consumed by orchestrator planning, formatting, and review calls.

Uncoordinated condition Each agent receives only its sub-task description, with no orchestrator intermediary. For sequential tasks, we still pass prior-stage output to the next agent as causal delivery, because ordered handoff is part of the task specification rather than a discretionary coordination layer. No orchestrator reviews, reconciles, or reformats outputs. We concatenate agent outputs sequentially under labeled section headers.

Validity evaluation GPT-4.1 mini acts as LLM-as-judge, running separately from the experiment agents. The evaluator receives the task-specific validity prompt plus the full final output and reasons step-by-step against each criterion before writing VALID or INVALID on the final line. We track evaluation tokens separately and exclude them from the coordinated and uncoordinated token totals. For non-monotonic tasks, the validity constraints are arithmetically verifiable (budget sums, story point totals, production hours, headcount), so we supplement the LLM judge with a programmatic validator as described below. These simulations are intended as controlled illustrations of the framework in an AI setting, not as direct observations of organizational field behavior.

Programmatic robustness check Because the four non-monotonic tasks have hard numerical constraints (Table S6), we verify the LLM-as-judge verdicts with a programmatic validator that parses dollar amounts, story points, production hours, and headcount from each output and compares the totals against the task-specific thresholds. For uncoordinated outputs, the parser splits

Table S6: **Tasks used in the simulations with pre-specified validity criteria.**

Task	Type	Agents	Validity criterion
Strategy pillars	Pooled (M)	4	All four pillars present (Market Expansion, Technology Investment, Operational Excellence, Talent Development), each with a strategic objective and at least one initiative
Feature specs	Pooled (M)	4	All four module specs present (Authentication, Search, Reporting, Notifications), each with scope, API endpoints, and data model
Marketing content	Pooled (M)	4	Content present for all four channels (email, social media, blog, press release), each with headline, body, and CTA; no contradictions on product name or value proposition
Security audit	Pooled (M)	4	All four layer reports present (Network, Application, Data, Identity), each with at least one finding with severity rating and recommendation
Stage-gate review	Sequential (M-O)	3	All three stages present (technical, market, financial); market references technical findings; financial references both; no factual contradictions
Ticket escalation	Sequential (M-O)	3	All three tiers present (L1 Triage, L2 Resolution, L3 Specialist); L2 references L1 categories; L3 references L2 resolution attempt; no contradictions
Budget alloc.	Reciprocal (NM)	5	All five divisions represented; total allocation $\leq$ \$50,000,000
Backlog sprint	Reciprocal (NM)	4	Features from all four domains represented (UX, Backend, Infrastructure, Integrations); total story points $\leq$ 200
Production sched.	Reciprocal (NM)	4	All four products scheduled (Alpha, Beta, Gamma, Delta); Line 1 $\leq$ 40 hrs; Line 2 $\leq$ 40 hrs
Headcount alloc.	Reciprocal (NM)	5	All five departments have allocations (Engineering, Sales, Customer Success, Marketing, Operations); total $\leq$ 25 positions

on agent section headers and sums each agent's stated total. For coordinated outputs, it locates the orchestrator's final allocation table.

The programmatic validator agrees with the LLM judge on 100% of uncoordinated non-monotonic records (120/120 across all three models), confirming that independent agents invariably violate shared resource constraints. For coordinated non-monotonic records, agreement is 92.5% (GPT-4.1 mini), 100% (Sonnet 4.5), and 59.5% among parsed Haiku 4.5 records (22/37, with 3 parse failures). All coordinated disagreements are cases where the LLM judge evaluated initial agent requests rather than the orchestrator's reconciled allocation, meaning the programmatic validator is more accurate on these records. No disagreement changes the paper's qualitative finding.

Token overhead ratios Table S7 reports the coordinated/uncoordinated token ratio for each task and model. These are the empirical c values used in the Coordination Tax.

Table S7: **Coordinated/uncoordinated token ratio (c) by task and model.** These are the empirical c values used in the Coordination Ceiling (Equation 1).

Task	Type	GPT-4.1-mini	Haiku 4.5	Sonnet 4.5
Strategy pillars	M	3.1×	2.8×	2.9×
Feature specs	M	2.5×	2.7×	2.6×
Marketing content	M	3.3×	4.1×	3.9×
Security audit	M	3.2×	2.7×	2.5×
Stage-gate	M-O	2.3×	2.7×	2.6×
Ticket escalation	M-O	2.9×	2.9×	2.8×
Budget alloc.	NM	3.6×	3.1×	3.4×
Backlog sprint	NM	3.5×	3.2×	3.4×
Production sched.	NM	4.4×	4.3×	4.2×
Headcount alloc.	NM	2.9×	3.2×	3.0×

9 Sensitivity Analysis

Decomposition Robustness

The observed monotonicity rate may partly reflect how tasks are decomposed, not only the tasks themselves. To assess sensitivity to reasonable alternative decompositions, we reclassified a small sample of ten APQC tasks spread across APQC categories: five originally labeled monotone or ordering-only (M/M-O), and five originally labeled non-monotone (NM). The reclassification used the same formal definitions and heuristic tests as the main classification. For each task, we used an alternative decomposition that varied one of three dimensions: whether timing constraints sit inside the specification, whether shared-resource pools are explicit or absorbed into local budgets, or whether downstream review is additive or revisionary. We then reapplied the classification rule and counted a label flip only when a task moved between the monotone/ordering-only group and the non-monotone group.

The sensitivity analysis showed that seven of the ten tasks retained their monotonicity status under the alternative decomposition (Table S8). Three crossed the monotonicity boundary: one ordering-only review task becomes non-monotone when review is modeled as revisionary, and two resource-allocation tasks become monotone when the shared pool is divided into local budgets in

Table S8: **Sensitivity analysis for ten APQC tasks under alternative decompositions.** Each task is reclassified under one alternative reasonable decomposition that changes the stated modeling assumption. Boundary flips count only changes between monotone or ordering-only labels (M/M-O) and non-monotone labels (NM), because that boundary determines the prevalence estimate and Coordination Tax.

Task	Original	Dimension	Alternative decomposition	New	Reason
Conduct competitive landscape analysis	M	Review	Central reviewer appends comments without requiring regional sections to be revised	M	Additive review enlarges the report without invalidating prior sections
Develop and cascade KPIs from strategy	M-O	Timing	Treat cascade order as implementation protocol rather than correctness condition	M	Independent KPI drafts can be merged when the final spec does not require causal handoff
Stage-gate product development review	M-O	Review	Model gates as revisionary approvals that can reject earlier specifications	NM	Later gate decisions can make prior stage outputs inadmissible
Process service request through fulfillment pipeline	M-O	Timing	Treat fulfillment stages as independently populated fields in a shared case record	M	Outputs accumulate by field when no stage can retract another's contribution
Audit regulatory compliance by domain	M	Review	Central reviewer appends issue notes without changing domain findings	M	Domain findings remain admissible as review notes are added
Allocate strategic investment budget	NM	Resource	Replace a shared investment pool with preallocated divisional envelopes	M	Local envelopes remove cross-division budget conflict
Prioritize product backlog with fixed capacity	NM	Review	Allow product owners to revise feature requests after seeing other teams' proposals	NM	Revisionary updates retract prior admissibility while sprint capacity remains binding
Schedule production across shared lines	NM	Timing	Treat sequencing rules as local, while shared production lines remain global constraints	NM	Competing jobs can still exceed shared line capacity
Assign service technicians to work orders	NM	Resource	Partition technicians by region before assignment	M	Regional pools remove competition for the same technician
Allocate operating budget across divisions	NM	Timing	Treat divisional budget proposals as concurrent submissions rather than a serial round-robin	NM	Concurrent proposals still compete for the same corporate budget

advance. These boundary crossings are substantively informative because they show that decomposition choices can sometimes introduce or remove non-monotonicity by changing where coordination constraints are represented. Of the seven stable tasks, two also shift from M-O to M when ordering is treated as an implementation protocol rather than a correctness condition, which does not affect the monotonicity rate. Taken together, most tasks retain the same monotonicity status under reasonable decomposition variation, and the 74% monotonicity rate remains decomposition-dependent rather than an invariant property of the tasks.

Scope of the Prevalence Estimates

The 74% monotonicity rate holds at APQC Levels 2–3 (team-level workflows). Reclassifying all borderline cases yields 60–74% monotonic. The lower bound (60%) assumes every borderline task is non-monotonic; the upper bound (74%) is the observed rate. Coarser decomposition pushes toward non-monotonic because bundling tasks together increases the chance of introducing shared constraints. Finer decomposition pushes toward monotonic because smaller sub-tasks are more likely to be independent.

The most dangerous failure mode is a task that appears monotonic but carries implicit non-monotonic constraints (e.g., unstated consistency requirements across independent reports). The reverse also occurs. Tasks that look non-monotonic may decompose into monotonic sub-tasks if resources are pre-partitioned. More broadly, the Bridge Theorem and Coordination Tax apply to any multi-agent setting where tasks can be decomposed and classified, but the prevalence parameter depends on the unit of analysis. APQC's 74% reflects team-level workflows that decompose naturally into multi-agent sub-tasks. O*NET's 42% reflects individual work activities that may or may not warrant multi-agent treatment. Software engineering, scientific research, and creative workflows would yield different distributions still.

References

[1] Tom J. Ameloot, Frank Neven, and Jan Van den Bussche. Relational transducers for declarative networking. *Journal of the ACM*, 60(2):15:1–15:38, 2013.

[2] Joseph M. Hellerstein. The coordination criterion. *arXiv preprint arXiv:2602.09435*, 2026.

[3] Leslie Lamport. Time, clocks, and the ordering of events in a distributed system. *Communications of the ACM*, 21(7):558–565, 1978.

[4] Joseph M. Hellerstein and Peter Alvaro. Keeping CALM: When distributed consistency is easy. *Communications of the ACM*, 63(9):72–81, 2020.

[5] Marc Shapiro, Nuno Preguiça, Carlos Baquero, and Marek Zawirski. Conflict-free replicated data types. In *Stabilization, Safety, and Security of Distributed Systems (SSS)*, pages 386–400. Springer, 2011.

[6] Jay R. Galbraith. Organization design: An information processing view. *Interfaces*, 4(3):28–36, 1974.

[7] Phanish Puranam, Marlo Raveendran, and Thorbjørn Knudsen. Organization design: The epistemic interdependence perspective. *Academy of Management Review*, 37(3):419–440, 2012.

[8] Bart Victor and Richard S. Blackburn. Interdependence: An alternative conceptualization. *Academy of Management Review*, 12(3):486–498, 1987.

[9] Joseph E. McCann and Diane L. Ferry. An approach for assessing and managing inter-unit interdependence. *Academy of Management Review*, 4(1):113–119, 1979.

[10] Henry Mintzberg. *The Structuring of Organizations*. Prentice-Hall, Englewood Cliffs, NJ, 1979.

[11] Phanish Puranam. *The Microstructure of Organizations*. Oxford University Press, Oxford, 2018.

[12] Herbert A. Simon. The architecture of complexity. *Proceedings of the American Philosophical Society*, 106(6):467–482, 1962.

[13] Lyra J. Colfer and Carliss Y. Baldwin. The mirroring hypothesis: Theory, evidence, and exceptions. *Industrial and Corporate Change*, 25(5):709–738, 2016.

[14] Ronald H. Coase. The nature of the firm. *Economica*, 4(16):386–405, 1937.

[15] Oliver E. Williamson. *The Economic Institutions of Capitalism*. Free Press, New York, 1985.

[16] William G. Ouchi. Markets, bureaucracies, and clans. *Administrative Science Quarterly*, 25(1):129–141, 1980.

[17] Ivan D. Steiner. *Group Process and Productivity*. Academic Press, New York, 1972.

[18] Jaeho Lee and Christos A. Makridis. Task-level automation and workers' skills in the age of AI. *Information Economics and Policy*, 64:101054, 2023.

[19] Thomas W. Malone and Kevin Crowston. The interdisciplinary study of coordination. *ACM Computing Surveys*, 26(1):87–119, 1994.

[20] Peter Bailis, Alan Fekete, Michael J. Franklin, Ali Ghodsi, Joseph M. Hellerstein, and Ion Stoica. Coordination avoidance in database systems. *Proceedings of the VLDB Endowment*, 8(3):185–196, 2015.

[21] Tyna Eloundou, Sam Manning, Pamela Mishkin, and Daniel Rock. GPTs are GPTs: Labor market impact potential of LLMs. *Science*, 384(6702):1306–1308, 2024.