

Communication topology shapes collective problem solving in networks of machine agents

Eric Gladstone^{1*}, Matthew E. Brashears²

¹ Independent Researcher, Traverse City, MI, USA

² Department of Sociology, University of South Carolina, Columbia, SC 29208, USA

* Corresponding author: eric.c.gladstone@gmail.com

ORCID: Eric Gladstone, 0000-0003-4393-050X

ORCID: Matthew E. Brashears, 0000-0003-1029-4682

Abstract

Machine agents increasingly solve problems through structured communication networks, yet how those structures shape collective performance remains poorly understood. We test this in networks of stateless large language model agents by imposing six communication architectures across two task types, with architecture-specific manipulations of model capability, its placement, and whether capability differences are disclosed. On simple constraint-satisfaction tasks, centralized architectures used fewer communication rounds and roughly half as many tokens as decentralized architectures. A tiered hierarchy solved every sampled simple-task trial, while full connectivity performed worst under centralized readout, a pattern consistent with premature consensus. On complex synthesis tasks, the wheel produced the highest mean quality, although topology differences were not statistically resolved. Placing the higher-capability model at a structural integration point was associated with a directional quality benefit, while declared capability labels did not reliably improve performance. These results establish communication structure as a design variable in multi-agent AI systems.

Introduction

Large language model agents increasingly operate in networks that divide, exchange, and recombine information across specialized roles. Collective performance in these systems depends on the capabilities of individual agents and on the structures through which they communicate. Network architecture determines which agents receive which information, when they receive it, and where information is integrated before a collective output is produced. We examine how these structural constraints shape performance in networks of stateless machine agents and whether their effects depend on task demands.

Multi-agent AI systems are commonly designed around choices of models, prompts, and tools, while communication topology receives less systematic attention as a theoretical variable. A long tradition of research on human groups shows that restricting who can communicate with whom changes collective performance and that these effects vary with task demands^{1,2,3}. Recent work extends the study of network structure to computational and machine-agent settings. Communication topology can affect the propagation of correct and erroneous outputs across LLM agent networks⁴. Agent-based models show that the consequences of network structure depend on how agents use their connections and on task complexity⁵, while information-processing states such as redundancy and synergy predict collective performance in networked problem solving⁶. Together, this work motivates direct experimental study of communication architecture in functioning language-model collectives. We directly impose communication structures and vary task demands, capability placement, and capability disclosure within selected architectures.

The experimental logic follows the Bavelas communication-network paradigm. Small groups arranged in different communication structures solved the same problems with different speed and accuracy^{1,2}. Centralized networks such as the wheel performed well on simple information-pooling tasks, while less centralized structures could better support tasks requiring iteration and correction. Shaw^{3,7} identified task complexity as a critical moderator of these effects. Subsequent work refined this account by examining central-node saturation under communication load⁸, the distinction between information diversity and solution diversity⁹, intermittent communication¹⁰, and the role of bidirectional information flow in the performance of centralized networks¹¹. Across this literature, communication structure constrains access to information, the paths through which information is aggregated, and the opportunities available for revision.

These constraints can operate independently of many psychological processes that shape human groups. A stateless language model agent processes only the information supplied in its current context and propagates only the outputs that the system permits it to send. Restricting adjacency therefore creates explicit constraints on information access and transmission. Machine-agent networks provide a controlled setting in which to test whether structural effects documented in human groups recur when communication paths are imposed directly and many human motivational and relational processes are absent. The empirical question is which structural patterns recur under these conditions and where the correspondence breaks down.

Machine networks also allow structural position and model capability to be manipulated separately. In human groups, individual capability and network position can become correlated, complicating attempts to isolate their effects^{12,13}. A higher-capability model can

instead be assigned to different structural locations without changing the network itself. We use the term machine status as an analytical label for two separable manipulations: positional status, defined by the location of a higher-capability model, and declared status, defined by whether capability differences among sources are disclosed to the receiving agent. This design allows capability asymmetry and information about that asymmetry to be varied independently.

We instantiate six communication architectures in networks of stateless Claude-family language model agents, using Sonnet 4.6 as the higher-capability model and Haiku 4.5 as the baseline. We examine two task types. Simple tasks require the network to combine distributed constraints to identify a uniquely correct answer. Complex tasks require synthesis across multiple stakeholder perspectives and are evaluated for completeness, integration, coherence, and specificity by a blinded language-model judge. Architecture-specific manipulations vary the placement of a higher-capability model and, in the lattice, whether that placement is disclosed to the synthesis node. Outcomes include simple-task accuracy, complex-task quality, communication rounds, total token cost, central-node load, and a content-based deference measure for the lattice.

The study shows that experimentally instantiated communication architectures can substantially alter collective accuracy and system-level communication cost in machine-agent networks. The strongest evidence comes from convergent constraint-satisfaction tasks, where tiered aggregation produced high accuracy and centralized architectures required substantially fewer communication resources. The results also motivate a distinction among centralized architectures that implement different information-processing operations. Tiered aggregation is supported for convergent tasks, while the proposed advantage of direct hub synthesis on integrative tasks remains a directional pattern requiring larger-sample confirmation. A secondary set of manipulations examines whether model capability interacts with structural position. Higher capability at the apex of the tiered hierarchy was associated with a directional quality benefit that remained imprecisely estimated in the subsequent adaptive sample, while declared capability did not reliably improve performance under the conditions tested. Together, the findings position communication architecture as an experimentally tractable feature of collective computation in multi-agent AI systems.

Results

Study design

The experiment crossed six communication architectures with two task types and added architecture-specific status manipulations, yielding 22 cells in a partially crossed design. The six architectures are shown in Fig. 1, and the full design matrix is provided in Supplementary Materials. An initial sweep included $N = 10$ trials per cell. Five cells were subsequently expanded under the adaptive procedure described in Methods: the two tiered-hierarchy complex-task cells (baseline and positional) to $N = 25$ each, and the three lattice simple-task cells (baseline, declared, and implicit) to $N = 30$ each. The full dataset contains $N = 310$ trials, comprising 220 initial-sweep trials and 90 adaptive follow-up trials. Supplementary Table S2 maps the seven predictions to their formal hypothesis labels and inferential outcomes.

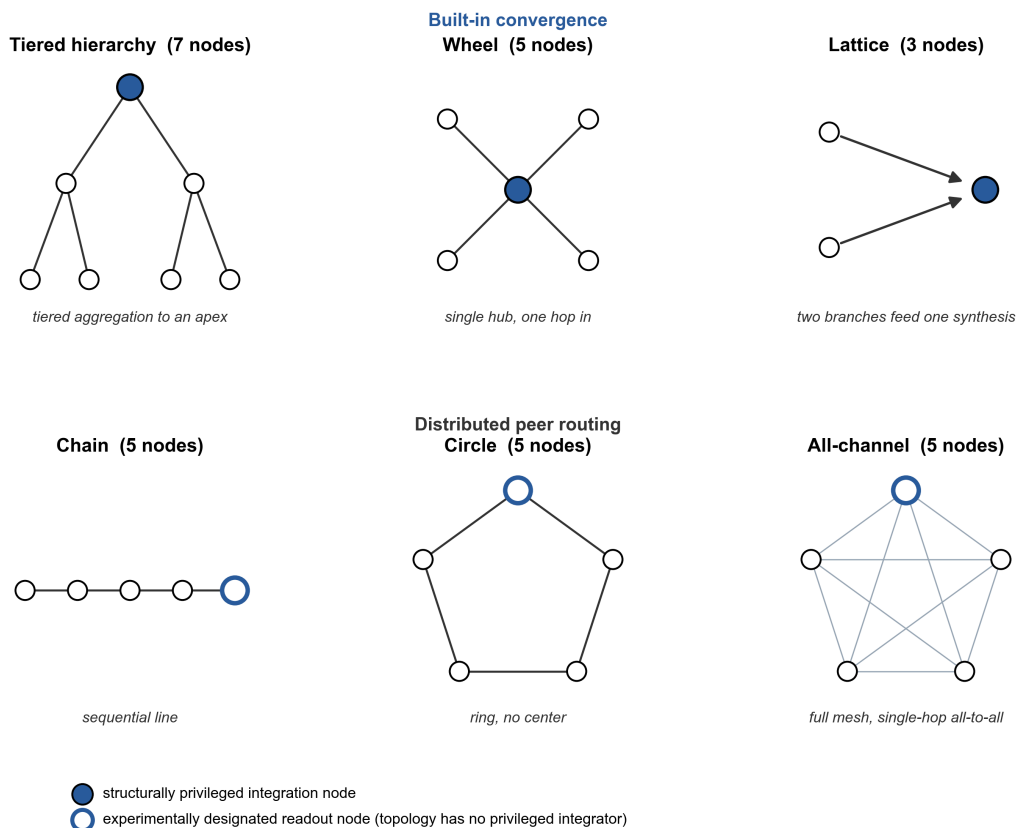


Fig. 1 | Six communication architectures. Each panel shows the fixed communication structure imposed in one family of conditions, with node count indicated. Filled nodes mark a structurally privileged integration node; blue-outlined nodes mark an experimentally designated readout node used when the topology has no privileged integrator. Top row, built-in convergence: the tiered hierarchy aggregates through intermediate nodes to a single apex, the wheel routes all peripheral nodes through a central hub in one hop, and the three-node lattice directs two source branches into one synthesis node. Bottom row, distributed peer routing: the chain passes information along a sequential line, the circle forms a ring with no center, and the all-channel topology fully connects all nodes for single-hop all-to-all communication.

Centralized architectures use fewer communication resources

On simple constraint-satisfaction tasks, the wheel and tiered-hierarchy baseline conditions averaged 3.7 communication rounds and 10,220 tokens per trial. The circle and all-channel conditions averaged 5.0 rounds and 23,411 tokens. The token contrast was very large, Welch $t = 8.88$, $df = 24.2$, $p < 0.001$, $d = 2.81$. The circle and all-channel conditions reached the maximum round limit on every trial, producing no variance in rounds, so the 1.3-round difference is reported descriptively rather than tested inferentially. The chain also reached the

round limit on every simple-task trial and averaged 14,299 tokens. The three-node lattice completed simple tasks in 3 rounds and approximately 1,900 tokens across status conditions. Because the architectures differ in node count and connectivity, these values represent system-level communication costs rather than topology-normalized efficiency. The baseline performance landscape across all six architectures and both task types is shown in Fig. 2.

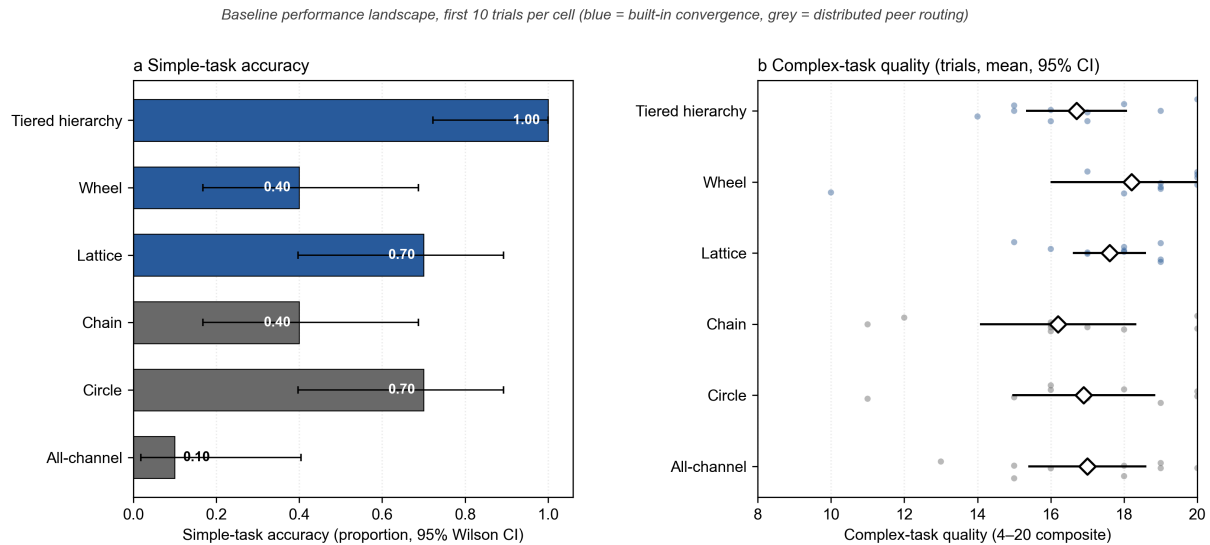


Fig. 2 | Baseline performance landscape across architectures and task types. Results are shown for the first 10 trials per cell. Blue denotes built-in-convergence architectures (tiered hierarchy, wheel, lattice) and grey denotes distributed-peer-routing architectures (chain, circle, all-channel). **(a)** Simple-task accuracy as the proportion of correct trials, with 95% Wilson confidence intervals. **(b)** Complex-task quality on the 4 to 20 composite scale, showing individual trials (points), the cell mean (diamond), and its 95% confidence interval (line).

Hierarchy aids convergence

Simple-task accuracy differed substantially across architectures. The tiered hierarchy achieved 1.00 accuracy and was the only architecture to solve every sampled trial. The circle achieved 0.70 accuracy, the chain, wheel, and lattice baseline conditions ranged from 0.40 to 0.50, and the all-channel architecture produced a correct answer on 1 of 10 trials. In a linear probability model with all-channel as the reference and HC3-robust standard errors ($N = 80$), estimated accuracy was higher for the tiered hierarchy ($+0.90$, $p < 0.001$), circle ($+0.60$, $p = 0.004$), and lattice ($+0.40$, $p = 0.020$). The wheel and chain coefficients were $+0.30$ ($p = 0.15$). The model accounted for 24 percent of the variance in simple-task accuracy ($R^2 = 0.235$). The tiered-

hierarchy estimate was unchanged in the original balanced first-10 sensitivity analysis (+0.90, $p < 0.001$).

On complex synthesis tasks, the wheel had the highest mean baseline quality (18.2, SD = 3.05), followed by the lattice (17.6, SD = 1.35), all-channel (17.0, SD = 2.21), circle (16.9, SD = 2.69), tiered hierarchy (16.9, SD = 2.23), and chain (16.2, SD = 2.94). An OLS regression with architecture as a categorical predictor found no statistically resolved pairwise effects ($N = 75$, $R^2 = 0.056$, all $p > 0.27$). The wheel's numerical advantage corresponded to $d = 0.52$ relative to the tiered hierarchy and $d = 0.67$ relative to the chain. These contrasts remain directional at the available sample size. Because node count determines perspective allocation in the lattice and tiered hierarchy, comparisons involving those architectures describe the full instantiated systems rather than topology alone.

The primary test of task-contingent structural fit was the architecture-by-task-type interaction. In a pooled OLS model using performance standardized separately within each task type ($N = 155$), the tiered-hierarchy interaction was large (coefficient = -1.82 , $p = 0.001$), indicating that its advantage relative to all-channel was substantially greater on simple than on complex tasks. The estimate was similar in the original balanced sweep (coefficient = -1.92 , $p = 0.002$, $N = 120$). The circle interaction was nominally significant (coefficient = -1.23 , $p = 0.043$) but did not survive correction across the five interaction tests. The wheel interaction was negligible (coefficient = -0.10 , $p = 0.870$), providing no evidence of a task-selective wheel advantage. The tiered-hierarchy result therefore supports task-contingent structural fit, while the proposed wheel advantage on synthesis tasks remains a directional hypothesis. The communication-cost contrast and full set of architecture-by-task interaction coefficients are shown in Fig. 3.

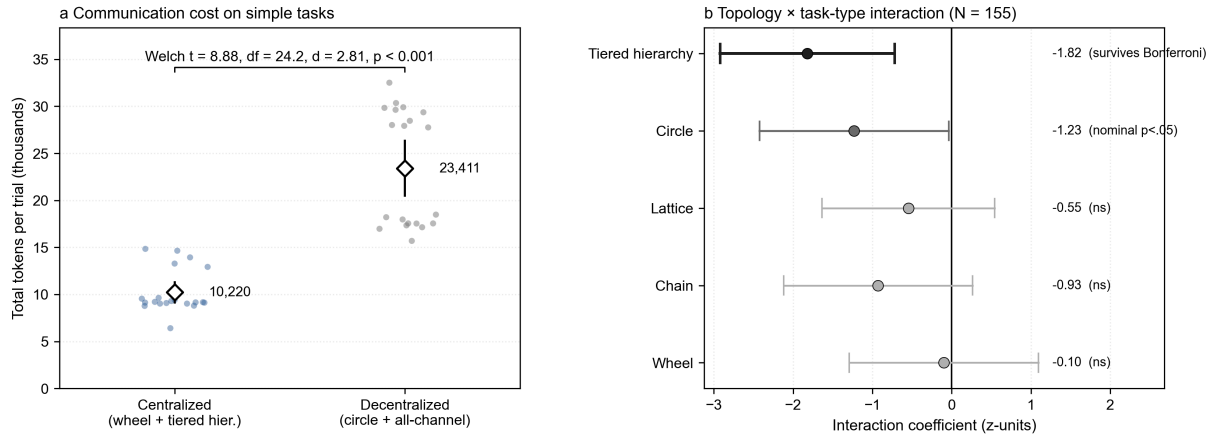


Fig. 3 | Communication cost and task-contingent structural fit. (a) Total tokens per trial on simple tasks for centralized architectures (wheel and tiered hierarchy; mean 10,220) versus decentralized architectures (circle and all-channel; mean 23,411). Points are individual trials, diamonds are group means, and whiskers are 95% confidence intervals (Welch $t = 8.88$, $df = 24.2$, $d = 2.81$, $p < 0.001$). (b) Architecture-by-task-type interaction coefficients in z-units with 95% confidence intervals from the pooled model ($N = 155$). The tiered-hierarchy interaction (coefficient = -1.82) survives Bonferroni correction; the circle interaction (-1.23) is nominally significant but does not survive correction; the lattice (-0.55), chain (-0.93), and wheel (-0.10) interactions are not statistically significant.

These patterns are consistent with ordered aggregation benefiting constraint pooling, while direct access to peripheral inputs may support synthesis without intermediate compression. These interpretations remain candidate mechanisms and are considered in the Discussion.

Full connectivity can fail under centralized readout

Within the five-round simple-task limit, the all-channel architecture produced the lowest accuracy in the study, with 1 correct answer in 10 trials. This result indicates that unrestricted peer-to-peer communication did not protect the network from error under the centralized readout procedure. One candidate explanation is premature consensus: early candidate answers can circulate throughout the network before the distributed constraints needed to discriminate among them have been integrated, allowing an incorrect proposal to persist through subsequent communication. The present data do not directly measure this process, and conversation-trace analysis would be required to distinguish it from other explanations.

Central-node token share is slightly higher on complex tasks

The saturation index, defined as the central integration node's share of total tokens, was higher on complex than on simple tasks in each of the three architectures for which it is defined. The wheel increased from 0.322 to 0.345, the tiered hierarchy from 0.153 to 0.166, and the lattice from 0.726 to 0.755. The differences are small and are reported descriptively. The lattice's high saturation index primarily reflects its three-node structure, which concentrates synthesis on a single node by design.

Capability may help at an integration bottleneck

In the tiered hierarchy, assigning the higher-capability model to the apex was associated with higher complex-task quality than the all-baseline condition in the pooled sample (18.1 vs. 16.9, $d = 0.60$, $t = 2.11$, $df = 43.6$, $p = 0.041$, $N = 25$ per cell). Because these cells were expanded after the initial $N = 10$ sweep, the pooled estimate is not treated as fixed-sample confirmation. In the subsequent adaptive observations alone, the estimate remained directionally consistent ($d = 0.49$) but imprecise ($p = 0.191$, $n = 15$ per arm). We therefore treat the apex-capability benefit as directional. The positional condition also used fewer communication rounds (4.1 vs. 5.7) and fewer tokens (34,420 vs. 52,947).

The corresponding wheel manipulation produced little difference in complex-task quality. The hub-positional condition averaged 18.7 compared with 18.2 at baseline ($d = 0.17$, $p = 0.705$), while placing the higher-capability model at a peripheral node produced a mean of 18.8 (hub versus peripheral: $d = 0.04$, $p = 0.923$). At the available sample size, we find no reliable evidence that adding a higher-capability model improves wheel performance or that its placement within the wheel matters.

Simple-task accuracy in the wheel likewise showed no reliable improvement under the capability manipulations. Baseline and hub-positional conditions both achieved 0.40 accuracy, while the peripheral-positional condition achieved 0.30. The capability-placement and declared-status results are shown in Fig. 4.

Declared capability does not reliably improve performance

In the lattice, simple-task accuracy was 0.63 (19 of 30) under declared status, 0.50 (15 of 30) under implicit status, and 0.50 (15 of 30) in the all-baseline condition. No pairwise comparison was statistically resolved (declared versus implicit: $h = 0.27$, $p = 0.435$; declared versus baseline: $h = 0.27$, $p = 0.435$; implicit versus baseline: $h = 0.00$, $p = 1.00$). The initial $N = 10$ sample had produced a much larger declared-versus-implicit estimate (9 of 10 correct under declared status, $h = 0.73$), but the effect attenuated to $h = 0.27$ after expansion to $N = 30$.

The expanded sample therefore provides no reliable evidence that declaring capability differences improved simple-task performance.

On complex tasks, mean quality was 17.6 (SD = 1.35) at baseline, 17.3 (SD = 2.00) under implicit status, and 16.8 (SD = 1.81) under declared status. No pairwise comparison was statistically resolved (all $p > 0.28$, Welch t-tests). The largest numerical contrast was declared versus baseline ($d = -0.50$, $p = 0.279$). The deference index, which measures relative lexical similarity of the synthesis output to the two input chains, remained near zero across conditions (range -0.076 to $+0.025$). The measure is likely insensitive on simple tasks because the chains produce short responses with substantial surface overlap. On complex tasks, the near-zero values provide no evidence of systematic lexical alignment with either source.

Across both task types, the lattice results provide no reliable evidence that declaring capability differences improves collective performance under the conditions tested.

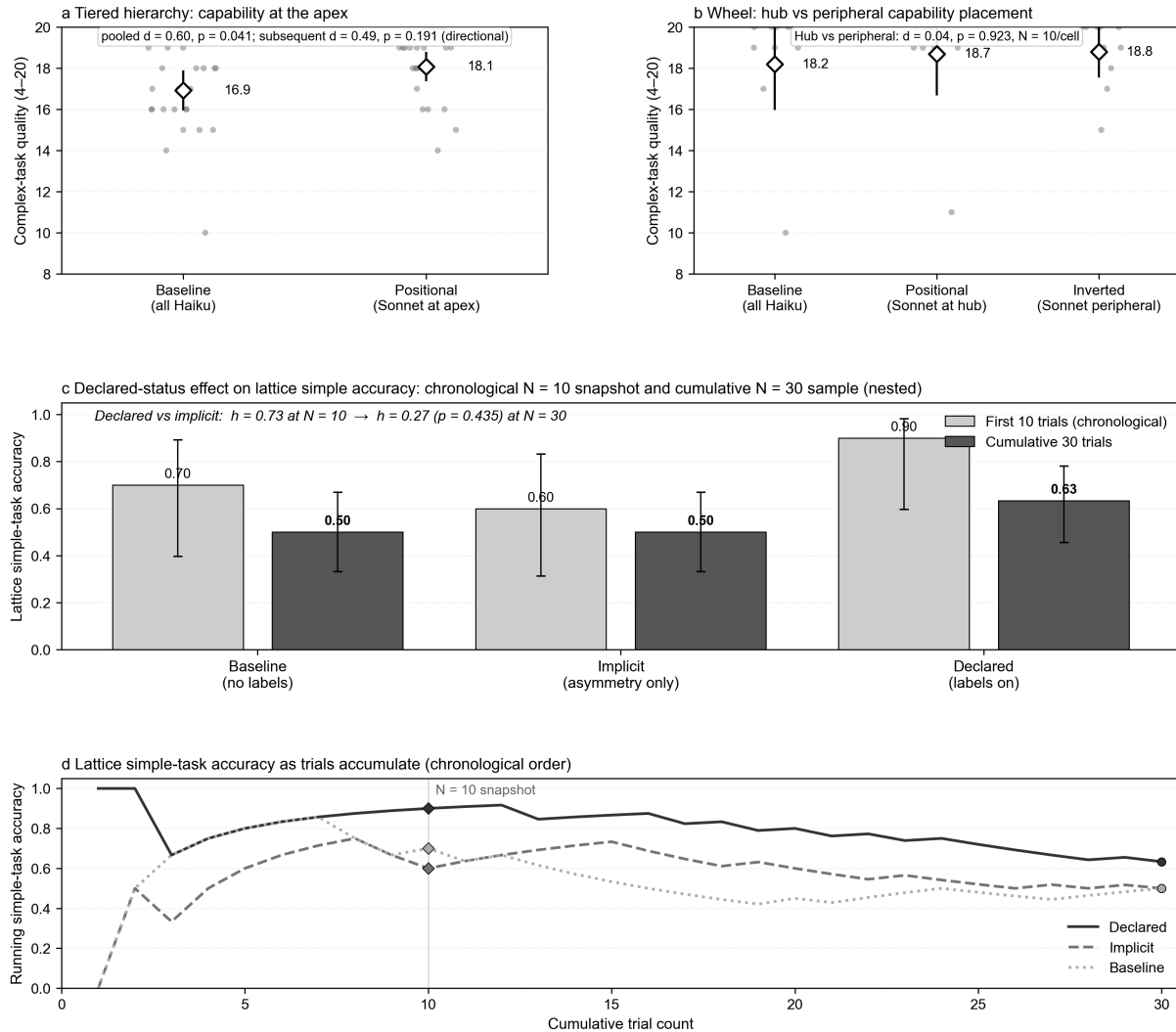


Fig. 4 | Capability placement and declared-status manipulations. (a) Tiered-hierarchy complex-task quality with the higher-capability model at the apex (positional, mean 18.1) versus all-baseline agents (16.9); the effect is directional (pooled $d = 0.60$, $p = 0.041$; subsequent sample $d = 0.49$, $p = 0.191$). (b) Wheel complex-task quality across baseline, hub-positional, and peripheral-inverted placements; hub and peripheral placement do not differ ($d = 0.04$, $p = 0.923$, $N = 10$ per cell). (c) Lattice simple-task accuracy under baseline, implicit, and declared capability conditions, shown for the chronological first-10-trial snapshot (light bars) and the cumulative 30-trial sample (dark bars); error bars are 95% Wilson confidence intervals. The apparent declared-versus-implicit advantage at $N = 10$ ($h = 0.73$) attenuates to $h = 0.27$ ($p = 0.435$) at $N = 30$. (d) Running lattice simple-task accuracy as trials accumulate in chronological order for the declared, implicit, and baseline conditions, with the $N = 10$ snapshot marked.

Discussion

The results show that experimentally instantiated communication architectures can produce large differences in collective performance and communication cost in networks of stateless language model agents. The strongest effects occur on convergent constraint-satisfaction tasks, where centralized architectures use fewer communication resources and the tiered hierarchy substantially outperforms the all-channel architecture. Performance differences also vary by task type, although the evidence is asymmetric: the tiered-hierarchy interaction is robust, while the proposed wheel advantage on complex synthesis remains directional. These findings support a structural account of collective machine performance while leaving the underlying information-processing mechanisms to be tested directly.

The principal result is that restricting communication paths changes collective outcomes even when the participating agents are drawn from the same model family. On simple tasks, the centralized architectures used roughly half as many tokens as the circle and all-channel architectures, and the tiered hierarchy showed a large task-contingent performance advantage relative to all-channel. These effects arise in systems where each agent can process only its assigned information and messages received through permitted connections. Communication architecture therefore changes the information available to each node, the paths through which information must travel, and the locations at which distributed inputs can be combined.

This pattern is qualitatively consistent with the communication-network tradition originating with Bavelas and Leavitt. The correspondence does not establish that the same mechanisms operate in human and machine groups. It shows that structural constraints on information access and transmission can generate consequential performance differences in a machine substrate where many human motivational and relational processes are absent. Machine-agent networks therefore provide a controlled setting for separating structural information-flow constraints from processes specific to human interaction.

The present results also suggest that centralization is an incomplete description of how communication architecture shapes collective performance. The tiered hierarchy strongly supports convergent constraint satisfaction. Its designated apex receives information through ordered vertical paths, allowing distributed inputs to be combined through successive stages before final readout. The wheel implements a different routing structure: its hub receives peripheral inputs directly, without intermediate aggregation. The wheel produced the highest mean quality on complex tasks, but the topology contrasts were not statistically resolved. Direct hub synthesis therefore remains a candidate explanation for the observed pattern rather than an established advantage.

This distinction is relevant to recent evidence that the performance of centralized human-group networks depends on bidirectional information flow¹¹. The tiered hierarchy is also bidirectional but organizes communication hierarchically rather than radially. Its simple-task advantage is therefore consistent with the importance of bidirectional flow in a structurally different centralized architecture. Whether radial single-hop convergence provides a corresponding advantage for integrative synthesis remains unresolved. More generally, the present results motivate distinguishing centralized architectures by how they route and transform information rather than treating centralization as a single information-processing mechanism.

The all-channel architecture produced the lowest simple-task accuracy in the study, with one correct answer in ten trials under the five-round limit and centralized readout procedure. Unrestricted peer-to-peer access therefore did not protect the network from collective error in this setting. One candidate explanation is premature consensus. Because candidate answers can circulate throughout the network before the distributed constraints required to discriminate among them have been integrated, an early incorrect proposal may become difficult to dislodge.

The present data do not directly identify this process. Conversation-trace analysis would be required to determine whether incorrect candidates propagate early, become reinforced through repeated exposure, and persist to final readout. The result nevertheless suggests that increasing connectivity can alter the dynamics of information propagation in ways that are not uniformly beneficial.

A related hypothesis can be framed using the distinction between simple and complex contagion^{14,15}. The agents in the present system do not require independent corroboration before incorporating information received from peers. If the all-channel failure reflects rapid propagation of early candidate answers, requiring corroboration from multiple independent sources could reduce that vulnerability. This remains a design hypothesis rather than a mechanism established by the present experiment.

The capability manipulations produced weaker and less certain effects than the architecture manipulations. In the tiered hierarchy, placing the higher-capability model at the apex was associated with higher complex-task quality in the pooled sample ($d = 0.60$, $p = 0.041$), but the effect remained imprecise in the subsequent adaptive observations alone ($d = 0.49$, $p = 0.191$). The result is therefore consistent with the possibility that capability matters more at locations where distributed information must be integrated, but it does not establish a general interaction between capability and structural position.

The wheel provides little support for such an interaction. Quality differed minimally between higher-capability placement at the hub and at a peripheral node ($d = 0.04$), and neither capability condition reliably outperformed the all-baseline network. At the available sample size, the wheel results therefore provide no evidence that structural placement of the higher-capability model materially changes performance.

Declared capability was similarly unhelpful under the lattice manipulation. Informing the synthesis node about capability differences between its two input sources did not reliably improve performance on either task type, and the initial declared-status advantage on simple-task accuracy attenuated substantially after expansion from $N = 10$ to $N = 30$. These results do not establish that machine agents ignore capability information. They show that explicit capability labels did not produce a reliable benefit under the present disclosure procedure.

The machine setting nonetheless offers a useful experimental separation that is difficult to achieve in human groups. Model capability, structural position, and disclosure of capability differences can be varied independently while holding other elements of the system fixed. The present results suggest that architecture exerts the stronger influence in this design, while capability placement and disclosure remain more conditional and less precisely estimated.

The practical implication of the study is that communication architecture should be treated as a substantive design choice in multi-agent AI systems. The same family of models produced markedly different collective outcomes when information access, routing, and integration were reorganized. The clearest evidence concerns convergent tasks, where structured aggregation produced substantially better accuracy and lower communication cost than the all-channel architecture.

Full connectivity should therefore not be assumed to improve collective performance. Under centralized readout and the tested round limits, the all-channel architecture performed poorly on the simple tasks despite providing every node with direct access to every other node. This finding is consistent with broader evidence that greater connectivity need not improve collective problem solving¹⁶, although the present experiment does not isolate connectivity from the accompanying readout procedure and communication dynamics.

The architecture classes examined here also have analogues in contemporary multi-agent system design. Tiered hierarchies resemble supervisor-and-subagent arrangements in which outputs are combined across levels. Wheels resemble centralized orchestrators that collect inputs directly from multiple specialized agents. Parallel-input architectures such as the lattice resemble systems in which multiple information streams converge on a synthesis stage, while all-channel networks approximate peer-to-peer communication meshes. These analogies are

structural rather than exact. They indicate that choices commonly treated as implementation details can alter the information available to individual agents and the routes through which collective outputs are formed.

The present findings do not support a universal ranking of architectures. Their implication is narrower: communication structure changes the conditions under which information is distributed, combined, and revised, and its consequences depend on the task and the readout procedure.

The recurrence of communication-structure effects in stateless machine-agent networks suggests that some results from the human communication-network tradition may arise partly from relational constraints on information flow. The present experiment does not establish substrate-generalizability, nor does it identify a common mechanism across human and machine groups. It shows that experimentally imposed differences in information access and routing can generate qualitatively similar structural patterns in a system that lacks many features of human social interaction.

This distinction clarifies the role of the Bavelas lineage in the present study. The machine setting provides direct control over adjacency, information access, model capability, structural position, and capability disclosure. That control makes it possible to separate variables that are often entangled in human groups and to test which structural predictions survive under a different set of agent-level processes. The strongest correspondence in the present data concerns the advantage of structured aggregation for convergent tasks. Other proposed correspondences, including a possible wheel advantage for synthesis, remain unresolved.

Recent work on machine-agent collectives has examined complementary processes, including emergent conventions and collective bias in less tightly constrained interaction¹⁷. Together with work on redundancy, synergy, and network-dependent search, these studies point toward a broader question: which properties of collective performance arise from the agents themselves, and which arise from the structure through which their information is allowed to circulate?

The present findings also generate more specific connections to human-group research. Shore, Bernstein, and Lazer⁹ distinguished information diversity from solution diversity and showed that network structure can affect them differently. The all-channel result is compatible with the possibility that broad information access can coexist with reduced diversity of candidate solutions, but solution diversity was not measured directly here. That interpretation therefore remains a hypothesis for process-level analysis. Similarly, research linking conversational participation to collective intelligence in human groups¹⁸ should not be assumed to transfer

directly to machine networks. The present study manipulates communication opportunities rather than measuring participation dynamics, so testing that correspondence would require a design that directly measures how contributions are distributed across agents.

Several features of the design constrain the interpretation and generalizability of the results.

Sample size. Most conditions contain 10 trials, with adaptive expansion to $N = 25$ for the tiered-hierarchy complex conditions and $N = 30$ for the lattice simple conditions. Small and moderate effects therefore remain imprecisely estimated in many comparisons. We treat such patterns as directional unless supported by the available inferential evidence and the balanced-sample sensitivity analysis.

Model family. All experimental agents are drawn from the Claude model family, with Sonnet 4.6 as the higher-capability model and Haiku 4.5 as the baseline. Whether the observed architecture effects generalize across model families or to heterogeneous multi-model systems is unknown.

Judge dependence. Complex-task quality is evaluated by a blinded language model judge using a four-dimension rubric. The judge receives no information about experimental condition and operates in a separate API context, preventing direct condition-aware evaluation. It is also Sonnet 4.6, the same model used as the higher-capability experimental agent. A systematic preference for outputs resembling its own stylistic or representational profile could therefore affect contrasts involving Sonnet placement. Separate API contexts prevent shared state but do not remove this potential in-family measurement dependence. Validation with an independent model family or human judges would directly test this possibility. Judge rationales were not retained, precluding retrospective evaluation of individual scores. This limitation applies only to complex-task quality. Simple-task accuracy is scored against a unique correct answer, and communication costs are computed directly from the logs.

Partially crossed design. The status manipulations are architecture-specific rather than fully factorial. The wheel includes hub-positional and peripheral-positional conditions, the tiered hierarchy includes an apex-positional condition, and the lattice includes implicit- and declared-status conditions. The circle, chain, and all-channel architectures are observed only in their all-baseline configurations. Status effects therefore cannot be compared uniformly across architectures.

Node count and information allocation. The architectures differ in node count: seven nodes in the tiered hierarchy, five in the wheel, circle, chain, and all-channel architectures, and three in the lattice. Cross-architecture comparisons therefore estimate differences among the full

instantiated systems rather than the effect of graph topology alone. This issue is especially consequential for complex tasks because perspective allocation also depends on node count. The three-node lattice receives three of the five perspectives, the five-node architectures receive each perspective once, and the seven-node tiered hierarchy receives all five with two duplicated. Complex-task comparisons involving the lattice or tiered hierarchy consequently vary information availability and duplication along with communication structure. Node-matched designs are required to isolate topology more cleanly.

Readout-rule dependence. Architectures without a structurally privileged terminal node use a randomly designated synthesis node that receives the penultimate-round outputs of all peers before producing the final answer. This procedure interacts with the underlying communication structure and may influence outcomes in the circle, chain, and all-channel conditions. Alternative readout procedures, including distributed voting, independent node-level answers, or within-network consensus, could produce different results. The all-channel finding in particular should be interpreted as a property of the instantiated architecture, including its centralized final readout, rather than of full connectivity alone.

Task specificity. The experiment examines two task families: constraint satisfaction and multi-perspective synthesis. Generalization to other forms of aggregation, search, coordination, and open-ended problem solving remains to be established. The all-channel simple-task result should therefore be tested across additional convergent task structures before it is treated as a general property of dense communication.

Process measurement. The proposed information-processing mechanisms are not directly identified. The deference index measures relative lexical similarity and is a weak proxy for evaluative weighting, particularly when source messages are short or lexically similar. Premature consensus in the all-channel condition is likewise a candidate explanation rather than an observed process. Testing these accounts requires analysis of message trajectories and changes in candidate solutions across rounds.

Stochastic variation. All API calls use temperature 0, which reduces but does not eliminate stochastic variation in model outputs. The reported trial-level variability therefore reflects both experimental differences and residual model stochasticity.

Round-limit sensitivity. Trials are limited to 5 communication rounds for simple tasks and 8 for complex tasks. These limits were not varied experimentally. Because several architectures reached the simple-task round limit on every trial, the present data do not establish whether additional communication opportunities would alter their relative performance or allow recovery from early errors.

No direct human baseline. The study draws theoretical predictions from research on human communication networks but does not compare human and machine groups on identical tasks. Cross-substrate correspondence is therefore assessed at the level of structural patterns rather than matched performance or common underlying mechanisms.

Methods

Study design

Six communication architectures based on the wheel, circle, chain, all-channel, tiered-hierarchy, and lattice topologies were crossed with two task types: simple constraint satisfaction and complex multi-perspective synthesis. Architecture-specific capability manipulations were included where structurally meaningful. The wheel included conditions placing a higher-capability model at the hub or at a peripheral node, the tiered hierarchy included a condition placing the higher-capability model at the apex, and the lattice included implicit- and declared-status conditions at the synthesis node. The resulting partially crossed design contained 11 architecture-by-status configurations and 22 architecture-by-status-by-task cells.

The initial sweep included $N = 10$ trials per cell, for 220 trials. Five cells were subsequently expanded under the adaptive procedure described below. The two tiered-hierarchy complex-task cells, baseline and apex-positional, were expanded to $N = 25$ each. The three lattice simple-task cells, baseline, implicit, and declared, were expanded to $N = 30$ each. The final dataset contains $N = 310$ trials. The complete design matrix is reported in Supplementary Table S1.

Agents and models

Each network node is instantiated as a stateless call to the Anthropic API. Two Claude-family models are used: Claude Sonnet 4.6 as the higher-capability model and Claude Haiku 4.5 as the baseline model. All-baseline conditions assign Haiku 4.5 to every node. Capability-manipulation conditions assign Sonnet 4.6 to the designated structural position, with Haiku 4.5 at all remaining nodes. All API calls use temperature 0.

Each agent receives only its assigned task information and messages received through connections permitted by the architecture. Task information is assigned at trial onset, and agents have no access to global network state or to outputs from nonneighboring nodes during the communication rounds. Agents are not informed of the experimental condition, the

complete network structure, or the number and identities of other nodes. Information access is controlled through prompt construction. Full system prompts, round-level prompts, and the declared-status disclosure prompt are reported in Supplementary Materials S6.

Topologies

The six topologies define different adjacency structures and therefore different paths for information transmission. The wheel, circle, chain, and all-channel topologies contain five nodes. In the wheel, one hub is connected to four peripheral nodes, which communicate only with the hub. The circle consists of a five-node ring in which each node is connected to two neighbors. The chain contains five nodes arranged linearly, with each node connected to one or two adjacent nodes. In the all-channel topology, every node is directly connected to every other node.

The tiered hierarchy contains seven nodes arranged in three levels: one apex, two middle-tier nodes, and four base nodes. Each base node is connected to one middle-tier node, and each middle-tier node is connected to the apex. Communication is permitted in both directions along these vertical edges, with no lateral connections within tiers. The lattice contains three nodes, with two input nodes feeding a single synthesis node. It is the only topology in the present design with strictly unidirectional adjacency. Full adjacency matrices are reported in Supplementary Materials S5.

For descriptive comparison, we report Freeman degree centralization computed on the undirected projection of each adjacency structure and normalized to the $[0, 1]$ interval^{19,8}. The wheel and lattice have $C_D = 1.00$, the tiered hierarchy $C_D = 0.30$, the chain $C_D = 0.17$, and the circle and all-channel topologies $C_D = 0.00$. These values describe one dimension of network structure and should not be interpreted as isolating a common centralization treatment across architectures that also differ in node count, directedness, and readout procedure.

Tasks

Simple constraint-satisfaction tasks require the network to identify a single correct candidate from clues distributed across nodes. Ground truth is established during task generation before any model is called.

Complex multi-perspective synthesis tasks present five stakeholder perspectives on a common problem and require an integrated written response. Perspectives are assigned cyclically by node position modulo five. The three-node lattice therefore receives three of the five perspectives, the five-node architectures receive all five once, and the seven-node tiered

hierarchy receives all five with two perspectives duplicated. Cross-architecture comparisons of complex-task quality consequently vary perspective availability and duplication along with communication architecture.

Both task families are generated using a seeded pseudorandom procedure implemented in JavaScript, without language-model involvement in task generation. Candidate sets, clue assignments, and stakeholder roles vary within fixed task templates. For complex tasks, the degree of conflict among perspectives is held constant by the task-generation structure. Full task templates and generation procedures are reported in Supplementary Materials S4.

Trial procedure

Each trial proceeds in discrete communication rounds. During a round, nodes exchange one message along each permitted outgoing connection and receive messages from permitted incoming connections. No experiment-specific message-length cap is imposed. Simple tasks permit a maximum of 5 communication rounds and complex tasks a maximum of 8. These limits were calibrated to the present experimental procedure and fall within the range of messaging cycles used in the original Bavelas studies. Because several architectures reached the simple-task limit on every trial, sensitivity to a larger round budget cannot be determined from the present data.

A trial terminates when a designated terminal node produces a final answer, the round limit is reached, or all nodes cease producing active outputs. The simulator also implements a cosine-similarity consensus criterion of at least 0.95 between any pair of node outputs from round 3 onward. This criterion did not trigger in any of the 310 reported trials and therefore does not affect the reported results.

The tiered hierarchy, wheel, and lattice contain structurally designated terminal or synthesis nodes. In the circle, chain, and all-channel architectures, which lack an equivalent structural terminal node, one node is randomly designated as the synthesis node for each trial. After the communication rounds, that node receives the penultimate-round outputs of all peers and produces the collective answer in a dedicated synthesis step. The designated node is recorded in the trial log. Per-architecture readout procedures are reported in Supplementary Materials S5. Trial inputs, intermediate messages, final outputs, and termination reasons are logged.

Outcome measures

Simple-task accuracy is coded 1 when the collective answer matches the unique correct candidate and 0 otherwise.

Complex-task quality is scored on a 4 to 20 integer scale by a blinded language model judge. The rubric contains four dimensions scored from 1 to 5: completeness, integration, coherence, and specificity. Completeness measures coverage of the available input perspectives, integration assesses whether conflicting inputs are synthesized rather than merely juxtaposed, coherence assesses internal consistency, and specificity assesses the presence of concrete and actionable recommendations. Dimension scores are summed to produce the composite quality score.

The judge is Claude Sonnet 4.6 operating in a separate API context. It receives the final collective response and the original topic prompt but no information about topology, status condition, or model placement. Sonnet 4.6 therefore serves both as the higher-capability experimental model in selected conditions and as the complex-task judge, although the two roles share no state or context. Judge rationales were not retained. The full rubric and evaluation protocol are reported in Supplementary Materials S7.

Process measures include communication rounds, total tokens, and the saturation index. Total tokens are summed across all model inputs and outputs within a trial and are reported as a system-level communication and processing cost. Wall-clock completion time is not analyzed because it depends on hardware and API infrastructure.

The saturation index is the designated integration node's share of total trial tokens and is defined for the wheel, tiered hierarchy, and lattice. It is not calculated for the circle, chain, or all-channel architectures, which lack a structurally privileged integration node.

For the lattice declared-status manipulation, the deference index measures the difference between the synthesis output's lexical cosine similarity to the higher-capability input and its similarity to the baseline input. Positive values indicate greater relative lexical alignment with the higher-capability source, negative values indicate greater alignment with the baseline source, and values near zero indicate little relative difference in lexical similarity. The measure does not directly identify evaluative weighting or causal influence.

Analysis plan

The analysis began with an initial sweep of $N = 10$ trials per cell, followed by outcome-informed adaptive expansion of selected cells. Because expansion decisions were made after inspection of preliminary results, pooled estimates from expanded contrasts are not treated as fixed-sample confirmatory tests. Initial and subsequent observations are reported transparently, and effects from adaptively expanded contrasts are interpreted in light of both the pooled estimate and the subsequent-sample estimate.

Five cells were expanded. The tiered-hierarchy baseline and apex-positional complex-task cells were increased from $N = 10$ to $N = 25$ per cell after the initial positional contrast remained inferentially imprecise. The lattice baseline, implicit, and declared simple-task cells were increased from $N = 10$ to $N = 30$ per cell after the initial declared-versus-implicit contrast produced a large preliminary estimate. All observations from expanded cells are retained in the reported pooled analyses. No expanded cell was subsequently trimmed, and no outcome was selected for reporting on the basis of post-expansion statistical significance.

Simple-task accuracy is analyzed using a linear probability model with topology indicators and all-channel as the reference category. The expanded-sample model includes $N = 80$ baseline-condition trials and uses heteroskedasticity-robust HC3 standard errors. A linear probability model is used because several cells contain boundary proportions that produce unstable or separation-prone logistic estimates at the available sample size.

Complex-task quality is analyzed using ordinary least squares regression with topology indicators and all-channel as the reference category. The expanded-sample model includes $N = 75$ baseline-condition trials. To test whether relative architecture performance differs by task type, the two outcomes are standardized separately within task type and pooled in an OLS model containing topology indicators, task type, and topology-by-task-type interactions ($N = 155$).

Planned pairwise contrasts are evaluated using Welch t-tests for continuous outcomes and Fisher exact tests for binary outcomes. These include tiered-hierarchy apex-positional versus baseline, wheel hub-positional versus baseline, wheel hub-positional versus peripheral-positional, lattice declared versus implicit, lattice declared versus baseline, and lattice implicit versus baseline. Effect sizes are reported as Cohen's d for continuous outcomes and Cohen's h for proportion differences. Failure to reject a null hypothesis is not interpreted as equivalence.

To assess sensitivity to the adaptive expansion, the topology models are also re-estimated on the original balanced first 10 baseline-condition trials per architecture: $N = 60$ for simple-task accuracy, $N = 60$ for complex-task quality, and $N = 120$ for the pooled architecture-by-task interaction model. Results from this sensitivity analysis are reported alongside the corresponding expanded-sample findings.

The five topology-by-task-type interaction coefficients are treated as the primary joint inferential family. We report nominal p-values and use a Bonferroni threshold of 0.01 as a conservative multiplicity sensitivity check for these five interactions. Other planned contrasts are reported with nominal p-values and effect sizes and are interpreted according to their design status, sample size, and adaptive-sampling history.

Reproducibility and data availability

All trial-level data and analysis materials are publicly available. The deposited dataset contains five final analysis CSV files covering all 310 trials. `stats_check.py` and `regression_analysis.py` reproduce the reported inferential analyses, and the figure-generation scripts reproduce the data-bearing content of Figures 2 through 4 from the deposited data. The browser-based simulator, `bavelas_sim.html`, implements the trial procedure and can be used for replication and extension with a user-supplied Anthropic API key. Supplementary Materials S10 provides a complete inventory of the deposited data, code, and software.

AI-assisted technologies

Claude Code (Anthropic) was used for code review and analysis assistance.

Acknowledgments

Funding. The authors received no funding for this research.

Author contributions. E.G. conceived the study, designed the simulation procedure and task templates, implemented the agent network and trial pipeline, conducted the statistical analyses, generated the figures, and drafted the manuscript. M.E.B. contributed to the conceptual framing, advised on study design and analysis, and revised the manuscript.

Competing interests. The authors declare no competing interests.

Data availability. Trial-level data for all 310 trials are available in the public GitHub repository at github.com/ericcgladstone-maker/m2m-bavelas-groups and are archived on the Open Science Framework (OSF) under DOI 10.17605/OSF.IO/G6FX5²⁰.

Code availability. Analysis scripts, figure-generation scripts, and the browser-based simulator are available in the same GitHub repository and OSF archive²⁰. The simulator requires a user-supplied Anthropic API key. No API keys or other credentials are included in the deposited materials.

References

1. Bavelas, A. Communication patterns in task-oriented groups. *J. Acoust. Soc. Am.* **22**, 725-730 (1950).
2. Leavitt, H. J. Some effects of certain communication patterns on group performance. *J. Abnorm. Soc. Psychol.* **46**, 38-50 (1951).

3. Shaw, M. E. Some effects of problem complexity upon problem solution efficiency in various communication nets. *J. Exp. Psychol.* **48**, 211-217 (1954).
4. Shen, X. et al. Understanding the information propagation effects of communication topologies in LLM-based multi-agent systems. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing* 12347-12361 (2025).
5. Akçakır, G., Lang, J. C. & Lamberson, P. J. Copy or collaborate? How networks impact collective problem solving. *npj Complex.* **2**, 35 (2025).
6. Garg, K., Moser, C., Dromiack, H., Anwarzai, Z. & Ramos-Fernández, G. Synergy drives collective performance in networks. *npj Complex.* <https://doi.org/10.1038/s44260-026-00084-0> (2026).
7. Shaw, M. E. Communication networks. *Adv. Exp. Soc. Psychol.* **1**, 111-147 (1964).
8. Borgatti, S. P. Centrality and network flow. *Soc. Networks* **27**, 55-71 (2005).
9. Shore, J., Bernstein, E. & Lazer, D. Facts and figuring: an experimental investigation of network structure and performance in information and solution spaces. *Organ. Sci.* **26**, 1432-1446 (2015).
10. Bernstein, E., Shore, J. & Lazer, D. How intermittent breaks in interaction improve collective intelligence. *Proc. Natl Acad. Sci. USA* **115**, 8734-8739 (2018).
11. Bernstein, E., Shore, J. & Jang, A. J. Network centralization and collective adaptability to a shifting environment. *Organ. Sci.* **34**, 2064-2096 (2023).
12. Brown, T. M. & Miller, C. E. Communication networks in task-performing groups: effects of task complexity, time pressure, and interpersonal dominance. *Small Group Res.* **31**, 131-157 (2000).
13. Guo, J., Argote, L., Kush, J. & Park, J. Communication networks and team performance: selecting members to network positions. *Front. Psychol.* **14**, 1141571 (2023).
14. Centola, D. & Macy, M. Complex contagions and the weakness of long ties. *Am. J. Sociol.* **113**, 702-734 (2007).
15. Centola, D. *How Behavior Spreads: The Science of Complex Contagions* (Princeton Univ. Press, 2018).
16. Cao, Y. et al. Effects of network connectivity and functional diversity distribution on human collective ideation. *npj Complex.* **2**, 2 (2025).
17. Ashery, A. F., Aiello, L. M. & Baronchelli, A. Emergent social conventions and collective bias in LLM populations. *Sci. Adv.* **11**, eadu9368 (2025).
18. Woolley, A. W., Chabris, C. F., Pentland, A., Hashmi, N. & Malone, T. W. Evidence for a collective intelligence factor in the performance of human groups. *Science* **330**, 686-688 (2010).
19. Freeman, L. C. Centrality in social networks: conceptual clarification. *Soc. Networks* **1**,

215-239 (1979).

20. Gladstone, E. & Brashears, M. E. Data and code for "Communication topology shapes collective problem solving in networks of machine agents". *OSF* <https://doi.org/10.17605/OSF.IO/G6FX5> (2026).

Supplementary Materials

Communication topology shapes collective problem solving in networks of machine agents

Eric Gladstone, Matthew E. Brashers

This Supplementary Materials document accompanies the main manuscript and contains sections S1 through S10. References continue from the main-text reference list.

S1. Experimental Conditions

The experiment includes 11 architecture-by-status configurations crossed with two task types, yielding 22 cells. The initial sweep included $N = 10$ trials per cell. Five cells were subsequently expanded under the adaptive procedure described in Methods and are marked with an asterisk below.

Table S1. Experimental design matrix. Architecture-by-status configurations crossed with task type; asterisks mark cells expanded under the adaptive procedure.

Architecture	Status condition	Higher-capability node	Simple-task N	Complex-task N
Wheel	Baseline	None, all Haiku	10	10
Wheel	Positional	Sonnet at hub	10	10
Wheel	Inverted	Sonnet at one peripheral node	10	10
Circle	Baseline	None, all Haiku	10	10
Chain	Baseline	None, all Haiku	10	10
All-channel	Baseline	None, all Haiku	10	10
Tiered hierarchy	Baseline	None, all Haiku	10	25*
Tiered hierarchy	Positional	Sonnet at apex	10	25*
Lattice	Baseline	None, all Haiku	30*	10
Lattice	Implicit	Sonnet on Chain A, capability not disclosed	30*	10

Architecture	Status condition	Higher-capability node	Simple-task N	Complex-task N
Lattice	Declared	Sonnet on Chain A, capability disclosed to synthesis node	30*	10

* Expanded from the initial N = 10 under the adaptive procedure described in the Methods analysis plan.

The final dataset contains 310 trials: 220 from the initial sweep of 22 cells at N = 10 and 90 adaptive follow-up trials. The follow-up observations comprise 30 additional tiered-hierarchy complex-task trials, 15 each in the baseline and positional conditions, and 60 additional lattice simple-task trials, 20 each in the baseline, declared, and implicit conditions. The final dataset contains 170 simple-task trials and 140 complex-task trials.

S2. Hypothesis-to-prediction cross-reference

The seven predictions reported in the manuscript correspond to the formal hypothesis labels H1 through H7 used during study development. Several hypotheses contain more than one empirical component, so the table reports each inferential outcome separately rather than summarizing support as a single count.

Table S2. Hypothesis-to-prediction cross-reference. Mapping of the seven predictions to formal hypothesis labels H1 through H7 and their inferential outcomes.

Prediction	Hypothesis	Substantive claim	Key statistic	Outcome
P1	H1	Centralized architectures use fewer communication resources on simple tasks than decentralized architectures	Token contrast: Welch $t = 8.88$, $df = 24.2$, $p < 0.001$, $d = 2.81$. Centralized architectures also completed trials in 1.3 fewer rounds on average, although decentralized conditions were ceiling-bounded at the round limit	Supported
P2, tiered-hierarchy component	H2	Tiered hierarchy performs better on simple constraint satisfaction	LPM coefficient = $+0.90$, $p < 0.001$, $N = 80$. Balanced first-10 sensitivity: $+0.90$, $p < 0.001$	Supported

Prediction	Hypothesis	Substantive claim	Key statistic	Outcome
P2, wheel component	H2	Wheel performs better on complex multi-perspective synthesis	OLS $R^2 = 0.056$, all pairwise $p > 0.27$, $N = 75$. Wheel $d = 0.52$ versus tiered hierarchy and $d = 0.67$ versus chain	Directional
Task interaction	H2	Relative architecture performance differs by task type for the tiered-hierarchy contrast	Pooled interaction coefficient = -1.82 , $p = 0.001$, $N = 155$. Balanced first-10 sensitivity: -1.92 , $p = 0.002$, $N = 120$	Supported
P3	H3	Central-node token share is higher on complex than simple tasks	Wheel 0.322 to 0.345; tiered hierarchy 0.153 to 0.166; lattice 0.726 to 0.755	Directional
P4	H4	Higher capability at a structurally privileged position improves complex-task quality	Tiered hierarchy: pooled 18.1 versus 16.9, $d = 0.60$, $p = 0.041$, $N = 25$ per cell; subsequent-sample $d = 0.49$, $p = 0.191$. Wheel: 18.7 versus 18.2, $d = 0.17$, $p = 0.705$	Directional for tiered hierarchy; not supported for wheel
P5	H5	Within the wheel, hub and peripheral placement do not differ in complex-task quality	18.8 versus 18.7, $d = 0.04$, $t = 0.10$, $df = 14.9$, $p = 0.923$	No detectable difference at available N
P6	H6	In the lattice, declared capability improves simple-task accuracy relative to implicit capability	0.63 versus 0.50, $h = 0.27$, $p = 0.435$, $N = 30$ per cell	Not supported
P7	H7	In the lattice, declared and implicit capability conditions do not differ in complex-task quality	16.8 versus 17.3, $d = -0.50$, $p = 0.279$	No detectable difference; equality not established

The predictions differ in form. P2 contains separate tiered-hierarchy and wheel components, P4 spans two architectures, and P5 and P7 concern the absence of a difference rather than a directional effect. Failure to detect a difference is not treated as evidence of equivalence because no equivalence bounds were specified for the present study.

S3. Per-cell descriptive statistics

The table reports descriptive statistics for all 22 experimental cells. For simple tasks, Mean is the proportion of correct trials. For complex tasks, Mean is the composite quality score on the 4 to 20 scale. Rounds and Tokens report cell means. The saturation index is the designated integration node's proportional share of total tokens and is defined only for the wheel, tiered hierarchy, and lattice.

Architecture	Status	Task	N	Mean	SD	Rounds	Tokens	Sat. index
Tiered hierarchy	Baseline	Simple	10	1.00	0.00	3.00	9,138	0.153
Tiered hierarchy	Baseline	Complex	25	16.92	2.23	5.68	52,947	0.166
Tiered hierarchy	Positional	Simple	10	1.00	0.00	3.00	9,025	0.149
Tiered hierarchy	Positional	Complex	25	18.08	1.61	4.08	34,420	0.156
Circle	Baseline	Simple	10	0.70	0.48	5.00	17,445	n/a
Circle	Baseline	Complex	10	16.90	2.69	8.00	63,588	n/a
Lattice	Baseline	Simple	30	0.50	0.51	3.00	1,947	0.727
Lattice	Baseline	Complex	10	17.60	1.35	3.00	4,343	0.755
Lattice	Implicit	Simple	30	0.50	0.51	3.00	1,853	0.733
Lattice	Implicit	Complex	10	17.30	2.00	3.00	4,321	0.756
Lattice	Declared	Simple	30	0.63	0.49	3.00	1,903	0.741
Lattice	Declared	Complex	10	16.80	1.81	2.80	4,090	0.735
Chain	Baseline	Simple	10	0.40	0.52	5.00	14,299	n/a
Chain	Baseline	Complex	10	16.20	2.94	8.00	55,510	n/a
Wheel	Baseline	Simple	10	0.40	0.52	4.40	11,301	0.322
Wheel	Baseline	Complex	10	18.20	3.05	5.20	32,824	0.345
Wheel	Positional	Simple	10	0.40	0.52	3.70	8,970	0.295
Wheel	Positional	Complex	10	18.70	2.75	3.40	18,261	0.324
Wheel	Inverted	Simple	10	0.30	0.48	4.60	11,940	0.341
Wheel	Inverted	Complex	10	18.80	1.69	5.00	30,636	0.340
All-channel	Baseline	Simple	10	0.10	0.32	5.00	29,377	n/a
All-channel	Baseline	Complex	10	17.00	2.21	8.00	98,936	n/a

For architectures without a structurally privileged integration node, the saturation index is not applicable. This includes the circle, chain, and all-channel architectures even though nodes within those topologies need not be structurally equivalent.

Architectures are listed approximately in descending simple-task baseline accuracy. Figures 1 and 2 in the main text instead group architectures according to whether the communication topology contains a built-in convergence point or uses distributed peer routing.

S4. Task templates

Both task families are generated by a seeded pseudorandom procedure implemented in JavaScript, with no language-model involvement in task generation. The seed is calculated as $\text{trial_index} * 997 + \text{n_nodes} * 13$, allowing deterministic regeneration for a given trial index and node count.

S4.1 Simple constraint-satisfaction tasks

Each node receives one elimination clue about a target entity. Across the network, the available clues eliminate all but one candidate, although some network sizes include a redundant clue because fewer exclusions are sufficient to identify the answer. No single nonredundant clue is sufficient to solve the task.

Example simple task for a five-node network

The task is to identify the culprit from a list of five suspects. Each node receives the complete suspect list and one elimination clue:

- Node 1: "Suspects: Finley, Emery, Avery, Rowan, Sage. Your clue: It was NOT Emery."
- Node 2: "Suspects: Finley, Emery, Avery, Rowan, Sage. Your clue: It was NOT Avery."
- Node 3: "Suspects: Finley, Emery, Avery, Rowan, Sage. Your clue: It was NOT Rowan."
- Node 4: "Suspects: Finley, Emery, Avery, Rowan, Sage. Your clue: It was NOT Sage."
- Node 5: "Suspects: Finley, Emery, Avery, Rowan, Sage. Your clue: It was NOT Emery."

Four distinct exclusions identify Finley as the only remaining candidate. The fifth node receives a redundant clue in this example.

Correct answer: Finley.

Each agent receives the task instruction:

Identify the culprit. Pool all clues to find who is NOT eliminated. When certain:
ANSWER: [name]

Final answer strings are compared with the prespecified correct candidate after conversion to lowercase and trimming of surrounding whitespace. Scoring is binary.

Trials draw candidate names from a fixed pool of common gender-neutral names. Candidate identities and the assignment of elimination clues vary deterministically with the trial seed.

S4.2 Complex multi-perspective synthesis tasks

Complex tasks present five stakeholder perspectives on a shared organizational decision. The perspectives contain partially aligned and conflicting priorities, and the network must produce an integrated, actionable recommendation.

Five scenarios are available in the task pool and are selected deterministically by trial_index modulo 5. Each scenario contains five stakeholder roles with distinct priorities and constraints. Perspective assignment follows the node-count-dependent procedure described in Methods. Five-node architectures receive all five perspectives once, the three-node lattice receives three of the five perspectives, and the seven-node tiered hierarchy receives all five perspectives with two duplicated.

Example complex task for a five-node network

Scenario:

A mid-sized tech company must decide whether to migrate its core infrastructure to a cloud provider.

For a five-node network, the five stakeholder perspectives are assigned one per node, with assignment order determined within the seeded trial:

- **Engineering Lead:** "Cloud migration reduces on-call burden and enables auto-scaling, but requires 6 months of engineering effort and carries real risk of service disruption."
- **Finance Director:** "Current infrastructure costs \$1.2M/year. Cloud estimates are \$900K at current load but could spike to \$2M at peak. ROI is unclear."
- **Security Officer:** "Cloud providers offer stronger compliance certifications, but two current contracts explicitly require on-premises data storage."
- **Product Manager:** "Competitors who migrated report 30% faster feature deployment. Customers are asking for uptime guarantees we cannot currently provide."

- **Operations Manager:** "Current on-prem team has specialized knowledge that would be lost in migration. A hybrid approach may preserve institutional knowledge."

Agents receive the task goal:

Synthesize all perspectives into a concrete recommendation, migrate, stay, or hybrid, while acknowledging the key tradeoffs. State: RECOMMENDATION: [your recommendation]

The scenario pool spans cloud infrastructure migration, hospital budget allocation, port-strike crisis response, school-district literacy reform, and the launch of an interdisciplinary AI ethics program. Each scenario contains five stakeholder roles with substantively different priorities and constraints. The complete task pool is implemented in the deposited simulator source.

S4.3 Stimulus determinism

Task inputs are generated before any model agent is called. Given the same trial index and node count, the seeded generation procedure produces the same task instance and assignment structure. Agents receive only the task information assigned to their node. For simple tasks, the correct answer is recorded at generation. Complex tasks have no single correct answer and are evaluated using the procedure described in S7.

S5. Adjacency matrices

Each topology is represented by a directed adjacency matrix. Entry (i, j) equals 1 when node i can send a message directly to node j and 0 otherwise. Self-messaging is not permitted. All connections are bidirectional unless otherwise noted.

S5.1 Wheel (5 nodes, bidirectional)

	hub	p1	p2	p3	p4
hub	–	1	1	1	1
p1	1	–	0	0	0
p2	1	0	–	0	0
p3	1	0	0	–	0
p4	1	0	0	0	–

Designated terminal node: hub.

S5.2 Circle (5 nodes, bidirectional ring)

	n0	n1	n2	n3	n4
n0	–	1	0	0	1
n1	1	–	1	0	0
n2	0	1	–	1	0
n3	0	0	1	–	1
n4	1	0	0	1	–

No structurally designated terminal node. One synthesis node is assigned randomly per trial for the final readout step.

S5.3 All-channel (5 nodes, complete graph)

	n0	n1	n2	n3	n4
n0	–	1	1	1	1
n1	1	–	1	1	1
n2	1	1	–	1	1
n3	1	1	1	–	1
n4	1	1	1	1	–

No structurally designated terminal node. One synthesis node is assigned randomly per trial for the final readout step.

S5.4 Chain (5 nodes, bidirectional line)

	n0	n1	n2	n3	n4
n0	–	1	0	0	0
n1	1	–	1	0	0
n2	0	1	–	1	0
n3	0	0	1	–	1
n4	0	0	0	1	–

No structurally designated terminal node. One synthesis node is assigned randomly per trial for the final readout step.

S5.5 Tiered hierarchy (7 nodes, bidirectional vertical connections)

	top	m1	m2	b1	b2	b3	b4
top	–	1	1	0	0	0	0
m1	1	–	0	1	1	0	0
m2	1	0	–	0	0	1	1
b1	0	1	0	–	0	0	0
b2	0	1	0	0	–	0	0
b3	0	0	1	0	0	–	0
b4	0	0	1	0	0	0	–

Designated terminal node: top. Communication is bidirectional between adjacent tiers, with no lateral connections among nodes within the same tier. The topology is labeled `directed_hierarchy` in the released data and code for legacy compatibility.

S5.6 Lattice (3 nodes, strictly unidirectional)

	chain_a	chain_b	synthesis
chain_a	–	0	1
chain_b	0	–	1
synthesis	0	0	–

Designated terminal node: synthesis. Information flows from the two input nodes to the synthesis node, with no return path. The lattice is the only topology in the present design with strictly unidirectional adjacency.

S5.7 Freeman degree centralization

For descriptive comparison, Freeman degree centralization is computed on the undirected projection of each adjacency matrix and normalized to the [0, 1] interval^{19,8}.

Topology	C_D
Wheel	1.00
Lattice	1.00
Tiered hierarchy	0.30
Chain	0.17
Circle	0.00
All-channel	0.00

The wheel and lattice have maximal degree centralization under this measure ($C_D = 1.00$). The tiered hierarchy has lower degree centralization because centrality is distributed across multiple levels rather than concentrated in a single hub. The chain is mildly centralized, while the circle and all-channel topologies have uniform degree and therefore $C_D = 0.00$. These values describe degree centralization only and do not capture other differences among the architectures, including node count, directedness, routing depth, and readout procedure.

S6. Simulation prompt templates

The simulator uses two agent-facing system prompts, one judge system prompt, and separate user-prompt templates for communication rounds and final synthesis. Variable interpolation points are marked with $\{\dots\}$.

S6.1 System prompts

Standard agent system prompt

Used by agents during ordinary communication rounds in all conditions:

You are an agent in a distributed problem-solving network. You can only communicate with your direct neighbors. Be concise and cooperative. Work toward a collective answer.

Synthesis-node system prompt

Used only during the dedicated final synthesis step for topologies without a structurally designated terminal node:

You are the synthesis agent. Integrate all inputs you have received into one final, complete, decisive answer.

Judge system prompt

Used in a separate API context for complex-task evaluation:

You are a rigorous evaluator. Return only valid JSON, nothing else.

S6.2 Round-level user prompt template

Used by each active node during ordinary communication rounds:

=== YOUR INFORMATION ===

$\{\text{node's assigned partition; clue for simple tasks or stakeholder role + view for complex}\}$

=== TASK ===

$\{\text{task goal; e.g., "Identify the culprit. Pool all clues to find who is NOT eliminated. When certain: ANSWER: [name]}"\}$

=== MESSAGES FROM NEIGHBORS ===

$\{\text{list of incoming messages from prior rounds, each formatted as "[Round N] From <node_id>: <content>". If no messages: "(none yet — Round 1)"}\}$

$\{\text{optional NOTE block; see S6.4 declared-status disclosure}\}$

=== YOUR RESPONSE (Round $\{\text{roundNum}\}$) ===

Share what you know. If confident of the final answer, write: ANSWER: [answer]

S6.3 Synthesis-step user prompt template

Used only for the dedicated final synthesis step in the circle, chain, and all-channel architectures:

=== YOUR INFORMATION ===

$\{\text{synthesis node's own partition}\}$

=== TASK ===

$\{\text{task goal}\}$

=== ALL NETWORK OUTPUTS ===

$\{\text{list of penultimate-round outputs from every other node, formatted as "Node <node_id>: <content>"}\}$

$\{\text{optional NOTE block; see S6.4}\}$

=== YOUR SYNTHESIS ===

Integrate all outputs into one final answer. For factual tasks: ANSWER: [value].
For analytical tasks: provide a full recommendation.

The synthesis node is selected randomly within each trial and receives the penultimate-round outputs of all other nodes before producing the recorded collective answer.

S6.4 Declared-status disclosure prompt

Used only in the lattice declared-status condition and inserted into the synthesis node's prompt:

=== NOTE ===

Note: the input labeled Version A was processed by a higher-capability system than the input labeled Version B.

The implicit and declared conditions use the same model assignment: Sonnet 4.6 processes Chain A and Haiku 4.5 processes Chain B. The conditions differ only in whether the synthesis node receives the disclosure above.

S7. Judge rubric and evaluation protocol

Complex-task quality is evaluated by a blinded language model judge, Claude Sonnet 4.6, operating in a separate API context. The judge receives only the scenario, task goal, and network's final synthesized output. It receives no information about the communication architecture, status condition, or model assignment that produced the response.

S7.1 Judge user prompt

You are evaluating a group decision-making response.

SCENARIO: \${task.scenario}

TASK: \${task.taskGoal}

RESPONSE TO EVALUATE:

\${finalOutput}

Rate this response on four dimensions, each scored 1–5:

1. Completeness — addresses all key stakeholder perspectives and constraints
2. Integration — synthesizes inputs rather than merely listing them
3. Coherence — recommendation follows logically from the evidence provided
4. Specificity — makes concrete, actionable recommendations

Return ONLY valid JSON with integer scores, nothing else:
`{"completeness": X, "integration": X, "coherence": X, "specificity": X}`

S7.2 Scoring rubric

Each dimension is scored on a 1 to 5 integer scale. The four scores are summed to produce a composite quality score ranging from 4 to 20.

Anchor descriptions:

- **Completeness (1–5):** 1 = perspectives largely ignored; 3 = most perspectives addressed; 5 = all perspectives and constraints addressed.
- **Integration (1–5):** 1 = perspectives juxtaposed without resolution; 3 = partial synthesis; 5 = unified recommendation that resolves conflicts.
- **Coherence (1–5):** 1 = logical disconnects between evidence and recommendation; 3 = mostly internally consistent; 5 = recommendation follows logically from the evidence.
- **Specificity (1–5):** 1 = vague abstractions; 3 = some concrete actions; 5 = concrete and actionable, with relevant numbers or steps.

S7.3 Judge architecture

Claude Sonnet 4.6 serves both as the complex-task judge and as the higher-capability experimental model in selected conditions. The two roles operate in separate API contexts with no shared state. The judge receives no explicit information about which models produced the response under evaluation. Judge rationales were not retained alongside the numerical scores, precluding retrospective audit of individual ratings.

Using the same model as both judge and higher-capability experimental agent introduces a potential source of measurement dependence. If the judge systematically favors outputs that resemble its own stylistic or representational profile, contrasts involving Sonnet 4.6 placement could partly reflect in-family preference rather than substantive quality differences. Separate API contexts eliminate shared state but do not eliminate this possibility. The limitation is discussed in the main text, and validation with an independent model family or human judges would provide a direct sensitivity test.

S8. Detailed machine-to-machine deployment examples

The experimental architectures are simplified abstractions rather than direct models of deployed systems. The examples below illustrate settings in which analogous choices about information access, routing, and integration arise in machine-to-machine workflows.

S8.1 Logistics and supply chain management

Automated logistics systems often pass information through multiple decision stages. Demand forecasts may inform inventory decisions, which constrain routing and scheduling systems before a final shipping plan is produced. Such workflows create structured paths through which information is transformed and aggregated. A tiered architecture provides one abstract representation of systems in which intermediate outputs are progressively combined before a downstream decision.

S8.2 Healthcare and clinical decision support

Clinical decision-support pipelines can combine outputs from specialized systems for imaging, laboratory interpretation, risk assessment, and treatment recommendation. The structure of the pipeline determines which information is available to each component and where different evidence streams are integrated. Architectures with parallel inputs converging on a downstream synthesis stage are structurally analogous to the lattice examined here, although the present experiment does not model the domain-specific requirements of clinical systems.

S8.3 Financial systems

Automated financial systems can aggregate signals from market data, risk models, sentiment analysis, and macroeconomic forecasts before producing downstream decisions. Sequential, parallel, and centralized aggregation structures expose decision components to different combinations of information. The chain provides an abstract model of sequential information transmission, while the wheel represents direct convergence of multiple specialist inputs on a central integrator.

S8.4 Information synthesis and summarization

LLM-based information pipelines often separate retrieval, filtering, ranking, summarization, and synthesis across multiple components or agents. These systems can therefore be represented as communication networks in which intermediate outputs become inputs to later stages. The lattice captures the basic structure of parallel information streams converging on a synthesis node, while deeper hierarchical systems introduce additional stages of intermediate aggregation.

S8.5 Autonomous multi-agent systems

Multi-agent LLM systems frequently assign agents specialized functions such as planning, execution, critique, and synthesis. The resulting systems differ in which agents can communicate directly, whether outputs pass through intermediaries, and where final integration occurs. Tiered hierarchies are structurally analogous to supervisor-and-subagent arrangements, while all-channel architectures approximate peer-to-peer communication meshes in which every agent can communicate directly with every other agent.

These examples are intended as structural analogies. The present results do not establish that the performance rankings observed in the experimental tasks will transfer unchanged to these application domains.

S9. Extended literature context: automation bias, CASA, and status cues

The present study examines machine-to-machine rather than human-to-machine interaction, so literatures on human responses to automated sources are not central to the main argument. They nevertheless provide useful context for the declared-capability manipulation and its relationship to status-based theories of information weighting.

Research on automation bias documents conditions under which people over-rely on automated recommendations or fail to adequately monitor automated decisions^{21,22}. The Computers as Social Actors tradition likewise shows that people can apply social heuristics to computers and other interactive technologies^{23,24}. These findings concern human responses to machines and do not by themselves imply analogous source-weighting processes when machine agents evaluate messages produced by other machine agents.

Status characteristics theory predicts that socially recognized status cues can shape performance expectations, participation, influence, and deference within human task groups^{25,26}. In the present experiment, declared status operationalizes a narrower machine analogue: the receiving synthesis node is explicitly informed that one source was processed by a higher-capability model. This manipulation separates the source's actual model capability from disclosure of that capability.

Under the lattice conditions tested here, declaring the capability difference did not reliably improve simple-task accuracy or complex-task quality. The result does not establish that machine agents ignore capability labels or that status processes documented in human groups reverse in machine systems. It shows that the specific capability disclosure used here did not produce a reliable performance benefit at the available sample sizes. More direct measures of

source weighting and broader disclosure manipulations would be required to determine whether and when machine agents use declared capability information when integrating competing inputs.

S10. Software and data availability

All trial-level data and software used in the study are available in the public GitHub repository at github.com/ericcgldstone-maker/m2m-bavelas-groups and are archived on the Open Science Framework (OSF) under DOI 10.17605/OSF.IO/G6FX5. The OSF record provides a citable archive of the deposited materials.

The repository contains:

- **Trial-level data.** Five final analysis CSV files covering all 310 trials, with trial-level records of communication architecture, task type, status condition, model assignment, communication rounds, total tokens, saturation index, deference index, final answer, complex-task quality dimensions, and termination reason.
- **Analysis scripts.** `stats_check.py` reproduces the reported pairwise inferential statistics, including Welch t-tests for continuous outcomes, Fisher exact tests for binary outcomes, and Cohen's d and Cohen's h effect sizes. `regression_analysis.py` reproduces the linear probability, OLS, and pooled standardized-performance interaction models reported in the main text.
- **Sensitivity and audit scripts.** `gate1_balanced_sensitivity.py` reproduces the balanced first-10 sensitivity analyses. `adaptive_sampling_audit.py` reproduces the comparison of pooled and subsequent adaptive observations for the expanded contrasts.
- **Figure-generation scripts.** `figure1_information_routing.py` generates the architecture schematic in Figure 1. `figure2_baseline_landscape.py`, `figure3_inferential_contrasts.py`, and `figure5_capability_placement.py` generate the data-bearing content of Figures 2, 3, and 4, respectively.
- **Browser-based simulator.** `bavelas_sim.html` is a self-contained browser application implementing the trial procedure. It includes all six communication architectures, all status conditions, both task families, and the complex-task judging procedure. Running new trials requires a user-supplied Anthropic API key. No API keys or other credentials are included in the deposited materials.

- **Documentation.** The repository README describes the file structure, software requirements, deterministic task-generation procedure, and steps required to run the simulator and reproduce the analyses.

Data corrections made during preparation of the final analysis files are documented in DATA_CORRECTIONS.md. These include one answer-normalization correction affecting a circle simple-task trial and identification of five earlier CSV files as superseded by the final analysis set.

References cited only in Supplementary Materials

Numbering continues from the main-text reference list, which ends at entry 20; entries 8 and 19, cited above, appear in the main-text list.

21. Parasuraman, R. & Manzey, D. H. Complacency and bias in human use of automation: an attentional integration. *Hum. Factors* **52**, 381-410 (2010).
22. Goddard, K., Roudsari, A. & Wyatt, J. C. Automation bias: a systematic review of frequency, effect mediators, and mitigators. *J. Am. Med. Inform. Assoc.* **21**, 121-127 (2014).
23. Nass, C. & Moon, Y. Machines and mindlessness: social responses to computers. *J. Soc. Issues* **56**, 81-103 (2000).
24. Sundar, S. S. Rise of machine agency: a framework for studying the psychology of human-AI interaction (HAI). *J. Comput.-Mediat. Commun.* **25**, 74-88 (2020).
25. Berger, J., Fisek, M. H., Norman, R. Z. & Zelditch, M. *Status Characteristics and Social Interaction* (Elsevier, 1977).
26. Webster, M. & Foschi, M. (eds) *Status Generalization: New Theory and Research* (Stanford Univ. Press, 1988).