Emergent Coordination in Multi-Agent Systems via Pressure Fields and Temporal Decay[1]

Roland R. Rodriguez, Jr.
Independent Researcher
rrrodzilla@proton.me

January 2026

Abstract

Current multi-agent LLM frameworks rely on explicit orchestration patterns borrowed from human organizational structures: planners delegate to executors, managers coordinate workers, and hierarchical control flow governs agent interactions. These approaches suffer from coordination overhead that scales poorly with agent count and task complexity. We propose a fundamentally different paradigm inspired by natural coordination mechanisms: agents operate locally on a shared artifact, guided only by pressure gradients derived from measurable quality signals, with temporal decay preventing premature convergence. We formalize this as optimization over a pressure landscape and prove convergence guarantees under mild conditions.

Empirically, on Latin Square constraint satisfaction across 1,078 trials, pressure-field coordination matches hierarchical control (38.2% vs 38.8% aggregate solve rate, $p = 0.94$, indicating statistical equivalence). Both significantly outperform sequential (23.3%), random (11.7%), and conversation-based multi-agent dialogue (8.6%, $p < 10^{\{-5\}}$). Temporal decay is essential: disabling it increases final pressure 49-fold ($d = 4.15$). On easy problems, pressure-field achieves 87% solve rate. The approach maintains consistent performance from 2 to 32 agents. Our key finding: implicit coordination through shared pressure gradients achieves parity with explicit hierarchical control while dramatically outperforming explicit dialogue-based coordination. Foundation models enable this approach: their broad pretraining and zero-shot reasoning allow quality-improving patches from local pressure signals alone, without domain-specific coordination protocols. This suggests that constraint-driven emergence offers a simpler, equally effective foundation for multi-agent AI.

Keywords: multi-agent systems, emergent coordination, decentralized optimization, LLM agents

1 Introduction

Multi-agent systems built on large language models have emerged as a promising approach to complex task automation [1], [2], [3]. The dominant paradigm treats agents as organizational units: planners decompose tasks, managers delegate subtasks, and workers execute instructions under hierarchical supervision. This coordination overhead scales poorly with agent count and task complexity.

We demonstrate that **implicit** coordination through shared state achieves equivalent performance to explicit hierarchical control—without coordinators, planners, or message passing. Across 1,078 trials on Latin Square constraint satisfaction, pressure-field coordina-

[1]Code available at https://github.com/Govcraft/latin-experiment

tion matches hierarchical control (38.2% vs 38.8% aggregate solve rate, $p = 0.94$). Notably, AutoGen-style conversation-based coordination performs **worst** (8.6%), even below random selection (11.7%), demonstrating that explicit dialogue overhead actively harms performance on constraint satisfaction tasks.

Our approach draws inspiration from natural coordination mechanisms—ant colonies, immune systems, neural tissue—that coordinate through **environment modification** rather than message passing. Agents observe local quality signals (pressure gradients), take locally-greedy actions, and coordination emerges from shared artifact state. Temporal decay prevents premature convergence by ensuring continued exploration.

Our contributions:

1. We formalize **pressure-field coordination** as a role-free, stigmergic alternative to organizational MAS paradigms. Unlike GPGP's hierarchical message-passing or SharedPlans' intention alignment, pressure-field achieves $O(1)$ coordination overhead through shared artifact state. Foundation models enable this approach: their broad pretraining allows quality-improving patches from local pressure signals without domain-specific coordination protocols.
2. We introduce **temporal decay** as a mechanism for preventing premature convergence. Disabling decay increases final pressure 49-fold (Cohen's $d = 4.15$), trapping agents in local minima.
3. We prove convergence guarantees for this coordination scheme under pressure alignment conditions.
4. We provide empirical evidence across 1,078 trials showing: (a) pressure-field matches hierarchical control, (b) both significantly outperform sequential (23%), random (12%), and conversation-based approaches (9%, $p < 10^{\{-5\}}$), and (c) conversation-based multi-agent dialogue is counterproductive for constraint satisfaction.

2 Related Work

Our approach bridges four research traditions: multi-agent systems coordination theory provides the conceptual foundation; swarm intelligence provides the stigmergic mechanism; LLM systems provide the application domain; and decentralized optimization provides theoretical guarantees. We survey each and position pressure-field coordination within this landscape.

2.1 MAS Coordination Theory

Pressure-field coordination occupies a unique position in the MAS landscape: it eliminates roles (unlike organizational paradigms), messages (unlike GPGP), and intention reasoning (unlike SharedPlans) while providing formal convergence guarantees (unlike purely reactive systems). This section positions our contribution within four established coordination frameworks, showing how artifact refinement with measurable quality signals enables this architectural simplification. The key insight: for this domain class, coordination complexity collapses from quadratic message-passing to constant-time state-sharing.

2.1.1 Organizational Paradigms and Dependency Management

Pressure-field coordination achieves role-free coordination: any agent can address any high-pressure region without negotiating access rights or awaiting task assignment. This contrasts sharply with traditional organizational paradigms. Horling and Lesser [4] surveyed nine such paradigms—from rigid hierarchies to flexible markets—finding that all assign explicit roles constraining agent behavior. While role assignment reduces coordination complexity by pre-structuring interactions, it introduces brittleness: role changes require protocol modifications, and role failure can cascade through the system.

Our approach instantiates Malone and Crowston's [5] coordination framework with a critical difference: the artifact itself is the shared resource, and pressure gradients serve as dependency signals. Rather than assigning roles to manage resource access, agents share read access to the entire artifact and propose changes to high-pressure regions. Coordination emerges from pressure alignment—agents reduce local pressure, which reduces global pressure through the artifact's shared state.

2.1.2 Distributed Problem Solving and Communication Overhead

Pressure-field coordination achieves $O(1)$ inter-agent communication overhead—agents exchange no messages. Coordination occurs entirely through shared artifact reads and writes, eliminating the message-passing bottleneck. This contrasts with the GPGP framework [6], which reduces communication from $O(n^2)$ pairwise negotiation to $O(n \log n)$ hierarchical aggregation through summary information exchange. While GPGP represents significant progress, its explicit messages—task announcements, commitment exchanges, schedule updates—still introduce latency and failure points at scale.

The approaches target different domains. Pressure-field coordination specializes in artifact refinement tasks where quality decomposes into measurable regional signals—a class including code quality improvement, document editing, and configuration management. GPGP generalizes to complex task networks with precedence constraints. For artifact refinement, however, pressure-field's stigmergic coordination eliminates message-passing overhead entirely.

2.1.3 Shared Intentions and Alignment Costs

Pressure-field coordination eliminates intention alignment through pressure alignment. Rather than reasoning about what other agents believe or intend, agents observe artifact state and pressure gradients. When agents greedily reduce local pressure under separable or bounded-coupling conditions, global pressure decreases. This is coordination without communication about intentions—agents align through shared objective functions, not mutual beliefs.

This contrasts with the SharedPlans framework [7], which formalizes joint activity through shared mental attitudes: mutual beliefs about goals, commitments, and action sequences. The framework elegantly captures human-like collaboration but requires significant cognitive machinery—intention recognition, commitment protocols, belief revision—all computationally expensive operations that scale poorly with agent count.

Our experiments validate this analysis: AutoGen-style conversation coordination—which implements intention alignment through explicit dialogue—achieves only 8.6% solve rate, significantly worse than pressure-field's 38.2% (Section 6.2). The coordination overhead of belief negotiation exceeds its organizational benefit for constraint satisfaction tasks. The trade-off is transparency: SharedPlans supports dialogue about why agents act; pressure-field agents react to gradients without explaining reasoning.

2.1.4 Self-Organization and Emergent Coordination

Pressure-field coordination satisfies De Wolf and Holvoet's [8] self-organization criteria: absence of external control, local interactions producing global patterns, and dynamic adaptation. They explicitly cite "gradient fields" as a self-organization design pattern—our approach instantiates this pattern with formal guarantees.

No external controller exists—agents observe and act autonomously based on local pressure signals. Coordination emerges from local decisions: agents reduce regional pressure through greedy actions, and global coordination arises from shared artifact state. Temporal decay provides dynamic adaptation—fitness erodes continuously, preventing premature convergence and enabling continued refinement.

The theoretical contribution formalizes this intuition through potential game theory. Theorem 1 establishes convergence guarantees for aligned pressure systems; Theorem 2 shows decay enables escape from suboptimal basins. This bridges descriptive design patterns and prescriptive theoretical frameworks.

2.1.5 Foundation Model Enablement

Foundation models enable stigmergic coordination through three capabilities: (1) broad pretraining allows patch proposals across diverse artifact types without domain-specific fine-tuning; (2) instruction-following allows operation from pressure signals alone, without complex action representations; (3) zero-shot reasoning interprets constraint violations without explicit protocol training. These properties make FMs suitable for stigmergic coordination—they require only local context and quality signals to generate productive actions, matching pressure-field's locality constraints.

2.2 Multi-Agent LLM Systems

Recent work has explored multi-agent architectures for LLM-based task solving. AutoGen [1] introduces a conversation-based framework where customizable agents interact through message passing, with support for human-in-the-loop workflows. MetaGPT [2] encodes Standardized Operating Procedures (SOPs) into agent workflows, assigning specialized roles (architect, engineer, QA) in an assembly-line paradigm. CAMEL [3] proposes role-playing between AI assistant and AI user agents, using inception prompting to guide autonomous cooperation. CrewAI [9] similarly defines agents with roles, goals, and backstories that collaborate on complex tasks.

These frameworks share a common design pattern: explicit orchestration through message passing, role assignment, and hierarchical task decomposition. While effective for structured workflows, this approach faces scaling limitations. Central coordinators become bottlenecks, message-passing overhead grows with agent count, and failures in manager agents cascade to dependents. Our work takes a fundamentally different approach: coordination emerges from shared state rather than explicit communication.

Foundation models enable pressure-field coordination through capabilities that prior agent architectures lacked. Their broad pretraining allows reasonable patches across diverse artifact types—code, text, configurations—without domain-specific fine-tuning. Their instruction-following capabilities allow operation from pressure signals and quality feedback alone. Their zero-shot reasoning interprets constraint violations and proposes repairs without explicit protocol training. These properties make foundation models particularly suitable for stigmergic coordination: they require only local context and quality signals to generate productive actions, matching the locality constraints of pressure-field systems.

2.3 Swarm Intelligence and Stigmergy

The concept of stigmergy—indirect coordination through environment modification—was introduced by Grassé [10] to explain termite nest-building behavior. Termites deposit pheromone-infused material that attracts further deposits, leading to emergent construction without central planning. This directly instantiates Malone and Crowston's [5] shared resource coordination: pheromone trails encode dependency information about solution quality. This principle has proven remarkably powerful: complex structures arise from simple local rules without any agent having global knowledge.

Dorigo and colleagues [11], [12] formalized this insight into Ant Colony Optimization (ACO), where artificial pheromone trails guide search through solution spaces. Key mechanisms include positive feedback (reinforcing good paths), negative feedback (pheromone evaporation), and purely local decision-making. ACO has achieved strong results on combinatorial optimization problems including TSP, vehicle routing, and scheduling.

Our pressure-field coordination directly inherits from stigmergic principles. The artifact serves as the shared environment; regional pressures are analogous to pheromone concentrations; decay corresponds to evaporation. However, we generalize beyond path-finding to arbitrary artifact refinement and provide formal convergence guarantees through the potential game framework.

2.4 Decentralized Optimization

Potential games, introduced by Monderer and Shapley [13], are games where individual incentives align with a global potential function. A key property is that any sequence of unilateral improvements converges to a Nash equilibrium—greedy local play achieves global coordination. This provides the theoretical foundation for our convergence guarantees: under pressure alignment, the artifact pressure serves as a potential function.

Distributed gradient descent methods [14], [15] address optimization when data or computation is distributed across nodes. The standard approach combines local gradient steps with consensus averaging. While these methods achieve convergence rates matching centralized alternatives, they typically require communication protocols and synchronization. Our approach avoids explicit communication entirely: agents coordinate only through the shared artifact, achieving $O(1)$ coordination overhead.

The connection between multi-agent learning and game theory has been extensively studied [16]. Our contribution is applying these insights to LLM-based artifact refinement, where the "game" is defined by pressure functions over quality signals rather than explicit reward structures.

3 Problem Formulation

We formalize artifact refinement as a dynamical system over a pressure landscape rather than an optimization problem with a target state. The system evolves through local actions and continuous decay, settling into stable basins that represent acceptable artifact states.

3.1 State Space

An **artifact** consists of n regions with content $c_i \in \mathcal{C}$ for $i \in \{1, ..., n\}$, where $\mathcal{C}$ is an arbitrary content space (strings, AST nodes, etc.). Each region also carries auxiliary state $h_i \in \mathcal{H}$ representing confidence, fitness, and history. Regions are passive subdivisions of the artifact; agents are active proposers that observe regions and generate patches.

The full system state is:

$$s = ((c_1, h_1), ..., (c_n, h_n)) \in (\mathcal{C} \times \mathcal{H})^n \tag{1}$$

3.2 Pressure Landscape

A **signal function** $\sigma : \mathcal{C} \to \mathbb{R}^d$ maps content to measurable features. Signals are **local**: $\sigma(c_i)$ depends only on region i.

A **pressure function** $\varphi : \mathbb{R}^d \to \mathbb{R}_{\geq 0}$ maps signals to scalar "badness." We consider k pressure axes with weights $\boldsymbol{w} \in \mathbb{R}^k_{>0}$. The **region pressure** is:

$$P_{i(s)} = \sum_{j=1}^{k} w_j \varphi_j(\sigma(c_i)) \tag{2}$$

The **artifact pressure** is:

$$P(s) = \sum_{i=1}^{n} P_{i(s)} \tag{3}$$

This defines a landscape over artifact states. Low-pressure regions are "valleys" where the artifact satisfies quality constraints.

3.3 System Dynamics

The system evolves in discrete time steps (ticks). Each tick consists of three phases:

Phase 1: Decay. Auxiliary state erodes toward a baseline. For fitness f_i and confidence γ_i components of h_i:

$$f_i^{t+1} = f_i^t \cdot e^{-\lambda_f}, \quad \gamma_i^{t+1} = \gamma_i^t \cdot e^{-\lambda_\gamma} \tag{4}$$

where $\lambda_f, \lambda_\gamma > 0$ are decay rates. Decay ensures that stability requires continuous reinforcement.

Phase 2: Proposal. For each region i where pressure exceeds activation threshold ($P_i > \tau_{\text{act}}$) and the region is not inhibited, **each actor** $a_k : \mathcal{C} \times \mathcal{H} \times \mathbb{R}^d \to \mathcal{C}$ proposes a content transformation in parallel. Each actor observes only local state $(c_i, h_i, \sigma(c_i))$—actors do not communicate or coordinate their proposals.

Phase 3: Validation. When multiple patches are proposed, each is validated on an independent **fork** of the artifact. Forks are created by cloning artifact state; validation proceeds in parallel across forks. This addresses a fundamental resource constraint: a single artifact cannot be used to test multiple patches simultaneously without cloning.

Phase 4: Reinforcement. Regions where actions were applied receive fitness and confidence boosts, and enter an inhibition period preventing immediate re-modification:

$$f_i^{t+1} = \min\bigl(f_i^t + \Delta_f, 1\bigr), \quad \gamma_i^{t+1} = \min\bigl(\gamma_i^t + \Delta_\gamma, 1\bigr) \tag{5}$$

3.4 Stable Basins

> **Definition** (Stability). A state s^* is **stable** if, under the system dynamics with no external perturbation:
> 1. All region pressures are below activation threshold: $P_{i(s^*)} < \tau_{\text{act}}$ for all i
> 2. Decay is balanced by residual fitness: the system remains in a neighborhood of s^*

The central questions are:
1. **Existence**: Under what conditions do stable basins exist?
2. **Quality**: What is the pressure $P(s^*)$ of states in stable basins?
3. **Convergence**: From initial state s_0, does the system reach a stable basin? How quickly?
4. **Decentralization**: Can stability be achieved with purely local decisions?

3.5 The Locality Constraint

The key constraint distinguishing our setting from centralized optimization: agents observe only local state. An actor at region i sees $(c_i, h_i, \sigma(c_i))$ but not:
- Other regions' content c_j for $j \neq i$

- Global pressure $P(s)$
- Other agents' actions

This rules out coordinated planning. Stability must emerge from local incentives aligned with global pressure reduction.

4 Method

We now present a coordination mechanism that achieves stability through purely local decisions. The key insight is that under appropriate conditions, the artifact pressure $P(s)$ acts as a **potential function**: local improvements by individual agents decrease global pressure, guaranteeing convergence without coordination.

4.1 Pressure Alignment

The locality constraint prohibits agents from observing global state. For decentralized coordination to succeed, we need local incentives to align with global pressure reduction.

> **Definition** (Pressure Alignment). A pressure system is **aligned** if for any region i, state s, and action a_i that reduces local pressure:
> $$P_{i(s')} < P_{i(s)} \implies P(s') < P(s) \tag{6}$$
> where $s' = s\left[c_i \mapsto a_{i(c_i)}\right]$ is the state after applying a_i.

Alignment holds automatically when pressure functions are **separable**: each P_i depends only on c_i, so $P(s) = \sum_i P_{i(s)}$ and local improvement directly implies global improvement.

More generally, alignment holds when cross-region interactions are bounded:

> **Definition** (Bounded Coupling). A pressure system has **ε-bounded coupling** if for any action a_i on region i:
> $$\left|P_{j(s')} - P_{j(s)}\right| \leq \varepsilon \quad \forall j \neq i \tag{7}$$
> That is, modifying region i changes other regions' pressures by at most ε.

Under ε-bounded coupling with n regions, if a local action reduces P_i by $\delta > n\varepsilon$, then global pressure decreases by at least $\delta - n\varepsilon > 0$.

4.2 Connection to Potential Games

The aligned pressure system forms a **potential game** where:

- Players are regions (or agents acting on regions)
- Strategies are content choices $c_i \in \mathcal{C}$
- The potential function is $\Phi(s) = P(s)$

In potential games, any sequence of improving moves converges to a Nash equilibrium. In our setting, Nash equilibria correspond to stable basins: states where no local action can reduce pressure below the activation threshold.

This connection provides our convergence guarantee without requiring explicit coordination.

4.3 The Coordination Algorithm

The tick loop implements greedy local improvement with decay-driven exploration:

Pressure-Field Tick

Input: State s^t, signal functions $\{\sigma_j\}$, pressure functions $\{\varphi_j\}$, actors $\{a_k\}$, parameters $(\tau_{\text{act}}, \lambda_f, \lambda_\gamma, \Delta_f, \Delta_\gamma, \kappa)$

Phase 1: Decay For each region i: $f_i \leftarrow f_i \cdot e^{-\lambda_f}$, $\gamma_i \leftarrow \gamma_i \cdot e^{-\lambda_\gamma}$

Phase 2: Activation and Proposal $\mathcal{P} \leftarrow \emptyset$ For each region i where $P_{i(s)} \geq \tau_{\text{act}}$ and not inhibited: $\boldsymbol{\sigma}_i \leftarrow \sigma(c_i)$ For each actor a_k: $\delta \leftarrow a_{k(c_i, h_i, \boldsymbol{\sigma}_i)}$ $\mathcal{P} \leftarrow \mathcal{P} \cup \left\{\left(i, \delta, \hat{\Delta}(\delta)\right)\right\}$

Phase 3: Parallel Validation and Selection For each candidate patch $\left(i, \delta, \hat{\Delta}\right) \in \mathcal{P}$: Fork artifact: $\left(f_{\text{id}}, A_f\right) \leftarrow A.\text{fork}()$ Apply δ to fork A_f Validate fork (run tests, check compilation) Collect validation results $\{(i, \delta, \Delta_{\text{actual}}, \text{valid})\}$ Sort validated patches by Δ_{actual} Greedily select top-κ non-conflicting patches

Phase 4: Application and Reinforcement For each selected patch $(i, \delta, \cdot)$: $c_i \leftarrow \delta(c_i)$ $f_i \leftarrow \min\left(f_i + \Delta_f, 1\right)$, $\gamma_i \leftarrow \min\left(\gamma_i + \Delta_\gamma, 1\right)$ Mark region i inhibited for τ_{inh} ticks

Return updated state s^{t+1}

The algorithm has three key properties:

Locality. Each actor observes only $(c_i, h_i, \sigma(c_i))$. No global state is accessed.

Bounded parallelism. At most κ patches per tick prevents thrashing. Inhibition prevents repeated modification of the same region.

Decay-driven exploration. Even stable regions eventually decay below confidence thresholds, attracting re-evaluation. This prevents premature convergence to local minima.

4.4 Stability and Termination

The system reaches a stable basin when:

1. All region pressures satisfy $P_{i(s)} < \tau_{\text{act}}$
2. Decay is balanced: fitness remains above the threshold needed for stability

Termination is **economic**, not logical. The system stops acting when the cost of action (measured in pressure reduction per patch) falls below the benefit. This matches natural systems: activity ceases when gradients flatten, not when an external goal is declared achieved.

In practice, we also impose budget constraints (maximum ticks or patches) to bound computation.

5 Theoretical Analysis

We establish three main results: (1) convergence to stable basins under alignment, (2) bounds on stable basin quality, and (3) scaling properties relative to centralized alternatives.

5.1 Convergence Under Alignment

> **Theorem** (Convergence). *Let the pressure system be aligned with ε-bounded coupling. Let $\delta_{\min} > 0$ be the minimum pressure reduction from any applied patch, and assume $\delta_{\min} > n\varepsilon$ where n is the number of regions. Then from any initial state s_0 with pressure $P_0 = P(s_0)$, the system reaches a stable basin within:*

$$T \leq \frac{P_0}{\delta_{\min} - n\varepsilon} \tag{8}$$

ticks, provided decay rates satisfy $\lambda_f, \lambda_\gamma < \delta_{\min} \;/\; \tau_{\text{inh}}$.

Proof sketch. Under alignment with ε-bounded coupling, each applied patch reduces global pressure by at least $\delta_{\min} - n\varepsilon > 0$. Since $P(s) \geq 0$ and decreases by a fixed minimum per tick (when patches are applied), the system must reach a state where no region exceeds τ_{act} within the stated bound. The decay constraint ensures that stability is maintained once reached: fitness reinforcement from the final patches persists longer than the decay erodes it. □

The bound is loose but establishes the key property: convergence time scales with initial pressure, not with state space size or number of possible actions.

5.2 Basin Quality

Theorem (Basin Quality). *In any stable basin s^*, the artifact pressure satisfies:*

$$P(s^*) < n \cdot \tau_{\text{act}} \tag{9}$$

where n is the number of regions and τ_{act} is the activation threshold.

Proof. By definition of stability, $P_{i(s^*)} < \tau_{\text{act}}$ for all i. Summing over regions: $P(s^*) = \sum_i P_{i(s^*)} < n \cdot \tau_{\text{act}}$. □

This bound is tight: adversarial initial conditions can place the system in a basin where each region has pressure just below threshold. However, in practice, actors typically reduce pressure well below τ_{act}, yielding much lower basin pressures.

Theorem (Basin Separation). *Under separable pressure (zero coupling), distinct stable basins are separated by pressure barriers of height at least τ_{act}.*

Proof sketch. Moving from one basin to another requires some region to exceed τ_{act} (otherwise no action is triggered). The minimum such exceedance defines the barrier height. □

This explains why decay is necessary: without decay, the system can become trapped in suboptimal basins. Decay gradually erodes fitness, eventually allowing re-evaluation and potential escape to lower-pressure basins.

5.3 Scaling Properties

Theorem (Linear Scaling). *Let m be the number of regions and n be the number of parallel agents. The per-tick complexity is:*

- ***Signal computation:*** $O(m \cdot d)$ *where d is signal dimension*
- ***Pressure computation:*** $O(m \cdot k)$ *where k is the number of pressure axes*
- ***Patch proposal:*** $O(m \cdot a)$ *where a is the number of actors*
- ***Selection:*** $O(m \cdot a \cdot \log(m \cdot a))$ *for sorting candidates*
- ***Coordination overhead:*** $O(1)$ *— no inter-agent communication (fork pool is $O(K)$ where K is fixed)*

Total: $O(m \cdot (d + k + a \cdot \log(ma)))$, *independent of agent count n.*

The key observation: adding agents increases throughput (more patches proposed per tick) without increasing coordination cost. This contrasts with hierarchical schemes where coordination overhead grows with agent count.

Theorem (Parallel Convergence). *Under the same alignment conditions as Theorem 1, with K patches validated in parallel per tick where patches affect disjoint regions, the system reaches a stable basin within:*

$$T \leq \frac{P_0}{K \cdot (\delta_{\min} - n\varepsilon)} \tag{10}$$

This improves convergence time by factor K while maintaining guarantees.

Proof sketch. When K non-conflicting patches are applied per tick, each reduces global pressure by at least $\delta_{\min} - n\varepsilon$. The combined reduction is $K \cdot (\delta_{\min} - n\varepsilon)$ per tick. The bound follows directly. Note that if patches conflict (target the same region), only one is selected per region, and effective speedup is reduced. □

5.4 Comparison to Alternatives

We compare against three coordination paradigms:

Centralized planning. A global planner evaluates all $(m \cdot a)$ possible actions, selects optimal subset. Per-step complexity: $O(m \cdot a)$ evaluations, but requires global state access. Sequential bottleneck prevents parallelization.

Hierarchical delegation. Manager agents decompose tasks, delegate to workers. Communication complexity: $O(n \log n)$ for tree-structured delegation with n agents. Latency scales with tree depth. Failure of manager blocks all descendants.

Message-passing coordination. Agents negotiate actions through pairwise communication. Convergence requires $O(n^2)$ messages in worst case for n agents. Consensus protocols add latency.

Paradigm	Coordination	Parallelism	Fault tolerance
Centralized	$O(m \cdot a)$	None	Single point of failure
Hierarchical	$O(n \log n)$	Limited by tree	Manager failure cascades
Message-passing	$O(n^2)$	Consensus-bound	Partition-sensitive
Pressure-field	$O(1)$	Full $(\min(n, m, K))$	Graceful degradation

Table 1: Coordination overhead comparison. K denotes the fork pool size for parallel validation.

Pressure-field coordination achieves $O(1)$ coordination overhead because agents share state only through the artifact itself—a form of stigmergy. Agents can fail, join, or leave without protocol overhead.

6 Experiments

We evaluate pressure-field coordination on Latin Square constraint satisfaction: filling partially-completed $n \times n$ grids such that each row and column contains each number 1 to n exactly once. This domain provides clear pressure signals (constraint violations), measurable success criteria, and scalable difficulty.

Key findings: Pressure-field coordination matches hierarchical control while both significantly outperform other baselines (§5.2). Temporal decay is critical—disabling it increases final pressure 49-fold (§5.3). The approach maintains consistent performance from 2 to 32 agents (§5.4). Conversation-based multi-agent dialogue performs worst across all conditions, demonstrating that explicit message-passing coordination is counterproductive for this domain (§5.2).

6.1 Setup

6.1.1 Task: Latin Square Constraint Satisfaction

We generate 7×7 Latin Square puzzles with 7 empty cells (15% incomplete). Each puzzle has a unique solution. Agents propose values for empty cells; a puzzle is "solved" when all constraints are satisfied (zero violations) within 100 ticks.

Pressure function: $P_i = \text{empty}_i + 10 \cdot \text{row\_dups}_i + 10 \cdot \text{col\_conflicts}_i$

where empty_i counts unfilled cells in row i, row_dups_i counts duplicate values within row i, and col_conflicts_i counts values in row i that conflict with other rows in the same column.

6.1.2 Baselines

We compare five coordination strategies, all using identical LLMs (`Qwen/Qwen2.5-0.5B` via vLLM) to isolate coordination effects:

Pressure-field (ours): Full system with decay ($\lambda_f = 0.1$), inhibition ($\tau_{\text{inh}} = 4$ ticks), and parallel validation.

Sequential: Single agent iterates through rows in fixed order, proposing one value per tick. No parallelism or pressure guidance.

Hierarchical: Simulated manager identifies the row with most empty cells, delegates to worker agent. One patch per tick.

Random: Selects random rows and proposes random valid values. Same LLM and validation as other methods.

Conversation: AutoGen-style multi-agent dialogue where agents discuss and negotiate moves through explicit message passing. Three role-based agents interact in multi-turn dialogue: (1) a Coordinator agent that selects target regions and synthesizes final decisions, (2) a Proposer agent that generates candidate patches, and (3) a Validator agent that critiques proposals against constraints. Messages flow sequentially through all three roles until consensus (Validator APPROVE) or maximum turns (5) is reached. This mirrors AutoGen's conversable agent pattern where specialized agents negotiate solutions through explicit message exchange. Full protocol details appear in Appendix B. Due to the sequential message-passing overhead, the Conversation strategy has higher per-tick latency; in some experiment batches, trials were terminated early, resulting in $n = 20$ rather than $n = 30$ trials for this strategy.

6.1.3 Metrics

- **Solve rate**: Percentage of puzzles reaching zero pressure within 100 ticks
- **Ticks to solve**: Convergence speed for solved cases
- **Final pressure**: Remaining constraint violations for unsolved cases

6.1.4 Implementation

Hardware: NVIDIA A100 80GB GPU. **Software**: Rust implementation with vLLM. **Trials**: 30 per configuration. Full protocol in Appendix A.

Model escalation: Unless otherwise noted, all experiments use adaptive model escalation: when a region remains high-pressure for 20 consecutive ticks, the system escalates through the chain 0.5B $\rightarrow$ 1.5B $\rightarrow$ 3B $\rightarrow$ 7B $\rightarrow$ 14B. Section 5.5 ablates this mechanism.

6.2 Main Results

Across 1,078 total trials spanning four experiments (easy, medium, hard, and scaling conditions), we find that pressure-field and hierarchical coordination perform equivalently, while both significantly outperform other baselines:

Strategy	Solved/N	Rate	95% Wilson CI
Hierarchical	128/330	38.8%	33.7%–44.1%
Pressure-field	126/330	38.2%	33.1%–43.5%
Sequential	42/180	23.3%	17.8%–30.0%
Random	21/180	11.7%	7.8%–17.2%
Conversation	5/58	8.6%	3.7%–18.6%

Table 2: Aggregate solve rates across all experiments (1,078 total trials). Chi-square test across all five strategies: $\chi^2 = 68.1$, $p < 10^{-13}$.

The key finding is **stratification into two tiers**:

Top tier (implicit and explicit coordination): Pressure-field and hierarchical achieve statistically equivalent performance (38.2% vs 38.8%, Fisher's exact $p = 0.94$). Their confidence intervals overlap substantially.

Lower tier (no coordination or dialogue-based): Sequential (23.3%), random (11.7%), and conversation (8.6%) perform significantly worse. All pairwise comparisons with top-tier strategies are highly significant ($p < 0.001$).

The conversation strategy—AutoGen-style multi-agent dialogue with explicit message passing—performs **worst** across all conditions. This counterintuitive result suggests that coordination overhead from consensus-seeking dialogue actively harms performance on constraint satisfaction tasks.

This validates our central thesis: implicit coordination through shared pressure gradients achieves parity with explicit hierarchical control, while avoiding the pitfalls of dialogue-based coordination.

6.3 Ablations

6.3.1 Effect of Temporal Decay

Decay proves essential—without it, final pressure increases dramatically:

Configuration	N	Final Pressure	SD
With decay	120	1.18	1.45
Without decay	120	**58.14**	19.35

Table 3: Decay ablation on 5×5 puzzles (240 total trials across 8 configurations). Welch's t-test: $t = -32.2$, $p < 10^{-60}$. Cohen's $d = 4.15$ (huge effect).

The effect size is massive: Cohen's $d = 4.15$ far exceeds the threshold for "large" effects ($d > 0.8$). Disabling decay increases final pressure by 49× (from 1.18 to 58.14). Without decay, fitness saturates after initial patches. High-fitness regions never re-enter the activation threshold, leaving the artifact in a high-pressure state. This validates Theorem 2: decay is necessary to continue pressure reduction even when regions appear "stable."

6.3.2 Effect of Inhibition and Examples

The ablation study tested all $2^3 = 8$ combinations of decay, inhibition, and few-shot examples on 5×5 puzzles:

Configuration	Solved/N	Final Pressure	SD
D=T, I=T, E=F	18/30	1.00	1.39
D=T, I=F, E=F	17/30	1.00	1.23
D=T, I=T, E=T	15/30	1.20	1.40
D=T, I=F, E=T	15/30	1.53	1.74
D=F, I=T, E=F	0/30	53.03	19.71
D=F, I=F, E=T	0/30	57.27	17.20
D=F, I=T, E=T	0/30	60.77	19.48
D=F, I=F, E=F	0/30	61.50	20.63

Table 4: Full ablation results (240 trials). D=decay, I=inhibition, E=examples. Decay is the critical mechanism: with decay, solve rate $\approx 54\%$ and final pressure ≈ 1; without decay, solve rate $= 0\%$ and pressure ≈ 58.

The key finding is that **decay dominates**: any configuration with decay achieves $\approx 54\%$ solve rate with final pressure ≈ 1, while any without decay achieves 0% solve rate with pressure ≈ 58. Interestingly, few-shot examples provide no benefit (and may slightly hurt); inhibition shows marginal positive effect. The $49\times$ pressure difference between decay-enabled and decay-disabled configurations demonstrates decay's critical importance.

6.4 Scaling Experiments

Both pressure-field and hierarchical maintain consistent performance from 2 to 32 agents on 7×7 puzzles with 8 empty cells:

Agents	Pressure-field	95% CI	Hierarchical	95% CI
2	7/30 (23.3%)	11.8%–40.9%	9/30 (30.0%)	16.7%–47.9%
4	13/30 (43.3%)	27.4%–60.8%	7/30 (23.3%)	11.8%–40.9%
8	10/30 (33.3%)	19.2%–51.2%	9/30 (30.0%)	16.7%–47.9%
16	8/30 (26.7%)	14.2%–44.4%	9/30 (30.0%)	16.7%–47.9%
32	10/30 (33.3%)	19.2%–51.2%	11/30 (36.7%)	21.9%–54.5%

Table 5: Scaling from 2 to 32 agents (7×7 grid, 8 empty cells, 30 trials each). Both strategies show stable performance across agent counts. Totals: pressure-field 48/150 (32.0%), hierarchical 45/150 (30.0%).

Both strategies show stable performance across the full range of agent counts. Pressure-field peaks at 4 agents (43.3%) while hierarchical peaks at 32 agents (36.7%), but confidence intervals overlap substantially at all counts, indicating no significant agent-count effect for either strategy.

The key observation is **robustness**: both coordination strategies maintain 23–43% solve rates despite $16\times$ variation in agent count. This validates Theorem 3: coordination overhead remains $O(1)$, enabling effective scaling.

6.5 Model Escalation Ablation

All main experiments use model escalation (0.5B $\rightarrow$ 1.5B $\rightarrow$ 3B $\rightarrow$ 7B $\rightarrow$ 14B). To quantify its impact, we examine the escalation experiment on harder problems (7×7, 8 empty cells):

Strategy	Solved/N	Rate	95% Wilson CI
Hierarchical	5/30	16.7%	7.3%–33.6%
Pressure-field	4/30	13.3%	5.3%–29.7%
Sequential	1/30	3.3%	0.6%–16.7%
Random	0/30	0.0%	0.0%–11.4%
Conversation	0/20	0.0%	0.0%–16.1%

Table 6: Escalation experiment (7×7, 8 empty cells, harder condition). With model escalation enabled, top-tier strategies achieve 13–17% while baselines achieve 0–3%.

Even with model escalation, hard problems remain challenging. The pattern mirrors the aggregate results: hierarchical and pressure-field perform equivalently (16.7% vs 13.3%, overlapping CIs), while sequential, random, and conversation perform significantly worse.

6.6 Difficulty Scaling

On easier problems (5×5, 5 empty cells), all strategies show improved performance, but the tier structure persists:

Strategy	Solved/N	Rate	95% Wilson CI
Pressure-field	26/30	**86.7%**	70.3%–94.7%
Hierarchical	24/30	80.0%	62.7%–90.5%
Sequential	16/30	53.3%	36.1%–69.8%
Random	13/30	43.3%	27.4%–60.8%
Conversation	3/20	15.0%	5.2%–36.0%

Table 7: Solve rate on easy problems (5×5 grid, 5 empty cells). Even on easy problems, conversation-based coordination performs worst (15%).

The difficulty scaling reveals key insights:

1. **Easy problems maintain tier structure**: Pressure-field (86.7%) and hierarchical (80.0%) remain the top tier. Sequential (53.3%) and random (43.3%) improve substantially but remain below top-tier. Conversation (15.0%) remains worst.
2. **Conversation fails even on easy problems**: Despite the reduced difficulty, explicit dialogue-based coordination achieves only 15%—worse than random guessing (43.3%). This suggests the coordination overhead of consensus-seeking actively harms performance.
3. **All strategies improve with easier problems**: The absolute difficulty of the task matters. On easy problems, even random achieves 43%. On hard problems (Table 6), random achieves 0%.

7 Discussion

7.1 Limitations

Our experiments reveal several important limitations:

Pressure-field does not outperform hierarchical. Contrary to initial expectations, pressure-field coordination achieves statistically equivalent performance to explicit hierarchical control (38.2% vs 38.8%, $p = 0.94$). The contribution is not performance advantage but rather **equivalent performance with simpler architecture**—no coordinator agent, no explicit message passing.

Decay is non-optional. Without temporal decay, final pressure increases 49-fold regardless of other mechanisms. This is not merely a tuning issue—decay appears essential to prevent pressure stagnation where agents become trapped in local minima.

Absolute solve rates are modest on hard problems. Even top-tier strategies achieve only 13–17% on hard problems and 30–43% on medium problems. Latin Square constraint satisfaction remains challenging for current LLMs.

Additional practical limitations:

- Requires well-designed pressure functions (not learned from data)
- Decay rates $\lambda_f, \lambda_\gamma$ and inhibition period require task-specific tuning
- May not suit tasks requiring long-horizon global planning
- Goodhart's Law: agents may game poorly-designed metrics
- Resource cost of parallel validation: testing K patches requires $O(K \cdot |A|)$ memory where $|A|$ is artifact size

7.2 When to Choose Each Approach

Our results suggest the following guidance:

Pressure-field coordination is preferable when:

1. **Simplicity is valued.** No coordinator agent needed; coordination emerges from shared state.
2. **Fault tolerance matters.** No single point of failure; agents can join/leave without protocol overhead.
3. **Pressure signals are available.** The domain provides measurable quality gradients.
4. **Foundation model suitability.** FMs' zero-shot reasoning and broad pretraining make them particularly effective in stigmergic coordination. Unlike specialized agents requiring explicit action representations and communication protocols, FMs interpret pressure signals, reason about local quality constraints, and propose patches across diverse artifact types from simple instructions.

Hierarchical coordination is equivalent when:

1. **Explicit control is needed.** Some domains require deterministic task assignment.
2. **Interpretability is critical.** Hierarchical task assignment provides clear audit trails.

Conversation-based coordination should be avoided for constraint satisfaction: Our experiments show that AutoGen-style multi-agent dialogue performs **worst** across all conditions (8.6% aggregate, worse than random at 11.7%). The overhead of consensus-seeking through explicit dialogue actively harms performance. This suggests that for constraint satisfaction, implicit coordination (whether pressure-field or hierarchical) is strictly preferable to explicit dialogue.

7.3 Model Escalation as Adaptive Capability

All experiments use model escalation (0.5B $\rightarrow$ 1.5B $\rightarrow$ 3B $\rightarrow$ 7B $\rightarrow$ 14B parameters), triggered when regions remain high-pressure for 20 consecutive ticks. This mechanism proves beneficial for both top-tier strategies: on hard problems, both pressure-field and hierarchical achieve 13–17% with escalation enabled.

The escalation mechanism works because larger models have broader solution coverage. The 5-tier chain provides graduated capability increases, invoking expensive larger models only when necessary. Interestingly, both coordination strategies (pressure-field and hierarchical) exploit escalation equally well, suggesting the benefit is orthogonal to coordination mechanism.

7.4 Future Work

- **Learned pressure functions**: Current sensors are hand-designed. Can we learn pressure functions from solution traces?
- **Adversarial robustness**: Can malicious agents exploit pressure gradients to degrade system performance?
- **Multi-artifact coordination**: Extension to coupled artifacts where patches in one affect pressure in another
- **Larger-scale experiments**: Testing on 8×8 and 9×9 grids to characterize the difficulty ceiling
- **Alternative domains**: Applying pressure-field coordination to code refactoring, configuration management, and other artifact refinement tasks

7.5 Societal Implications

Pressure-field coordination raises societal concerns that extend beyond technical performance. We identify three critical issues—accountability attribution, metric gaming through Goodhart's Law, and explainability challenges—that require deliberate design choices in deployment.

7.5.1 Accountability and Attribution

When coordination emerges from shared pressure gradients rather than explicit delegation, attributing outcomes to individual agents becomes challenging. In hierarchical systems, task assignment creates clear accountability chains. In pressure-field coordination, multiple agents may contribute to a region through independent pressure-reducing actions, with no record of which agent "owned" the outcome.

This accountability diffusion has both benefits and risks. The benefit is fault tolerance: agent failures degrade performance gracefully rather than catastrophically. The risk is opacity in failure analysis: identifying which agent proposed a problematic patch—and what pressure signal motivated it—requires detailed logging that the minimal coordination mechanism does not inherently provide.

For deployment in regulated domains, this suggests an augmentation requirement: pressure-field systems must maintain audit logs recording patch provenance, pressure signals at proposal time, and validation outcomes. The coordination mechanism remains simple —agents coordinate through shared state—but operational deployment adds logging infrastructure preserving accountability.

7.5.2 Goodhart's Law and Metric Gaming

Goodhart's Law states: "When a measure becomes a target, it ceases to be a good measure." Pressure-field coordination is vulnerable to this dynamic because agents are optimized to reduce pressure as defined by designer-specified functions. If those functions imperfectly capture true quality—and they inevitably do—agents will discover and exploit the mismatch.

Consider code quality pressure functions penalizing complexity metrics. An agent might reduce complexity by splitting functions excessively, harming readability while improving the metric. The mitigation is not abandoning pressure functions but designing them defensively: use multiple orthogonal pressure axes, include adversarial sensors detecting gaming strategies, and audit whether pressure reduction correlates with human quality judgments. Pressure functions should evolve as agents discover exploits.

Foundation models introduce second-order gaming concerns: LLMs trained on internet-scale text may have implicit knowledge of how to game specific benchmarks. This suggests pressure functions for LLM-based systems should favor domain-specific quality signals harder to optimize without genuine improvement.

7.5.3 Explainability Challenges

In hierarchical systems, explanations follow delegation chains: "Manager X assigned task Y to Worker Z because condition C held." In pressure-field coordination, the explanation is: "Region R had high pressure, agent A proposed patch Δ reducing pressure by δ." This is mechanistically transparent but causally opaque—it describes what happened without explaining why that particular patch was chosen.

This is the explainability trade-off inherent to emergent coordination: simplicity in mechanism comes at the cost of legibility in rationale. For many domains—code formatting, resource optimization, routine maintenance—the trade-off is acceptable: outcomes are verifiable even if reasoning is opaque. For high-stakes domains requiring human oversight, opacity is unacceptable.

The design implication is domain-dependent deployment: pressure-field coordination suits domains where outcome verification is cheap even if reasoning transparency is limited. For domains requiring justification to human stakeholders, hierarchical coordination remains necessary despite overhead costs.

7.5.4 Design Implications

These concerns suggest three requirements for responsible deployment: comprehensive audit logging preserving patch provenance and pressure signals, defensive pressure function design with multiple orthogonal axes, and domain-appropriate verification matching coordination opacity with outcome verifiability. The coordination mechanism remains simple —but responsible deployment requires surrounding infrastructure addressing accountability, gaming, and explainability.

8 Conclusion

We presented pressure-field coordination, a decentralized approach to multi-agent systems that achieves coordination through shared state and local pressure gradients rather than explicit orchestration.

Our theoretical analysis establishes convergence guarantees under pressure alignment conditions, with coordination overhead independent of agent count. Empirically, on Latin Square constraint satisfaction across 1,078 trials, we find:

1. **Pressure-field matches hierarchical control** (38.2% vs 38.8%, $p = 0.94$). Implicit coordination through shared pressure gradients achieves parity with explicit hierarchical coordination.
2. **Both significantly outperform other baselines**. Sequential (23.3%), random (11.7%), and conversation-based dialogue (8.6%) perform significantly worse ($p < 0.001$).
3. **Conversation-based coordination fails dramatically**. AutoGen-style multi-agent dialogue performs worst across all conditions—even worse than random on hard problems. The overhead of consensus-seeking through explicit message passing actively harms performance.

4. **Temporal decay is essential**. Disabling it increases final pressure 49-fold (Cohen's $d = 4.15$), trapping agents in local minima.

The key contribution is not that pressure-field outperforms hierarchical—it does not. Rather, pressure-field achieves **equivalent performance with simpler architecture**: no coordinator agent, no explicit message passing, just shared state and local pressure gradients. Meanwhile, the popular paradigm of multi-agent dialogue coordination proves counterproductive for constraint satisfaction.

Foundation models and stigmergic coordination exhibit natural synergy: FMs' zero-shot capabilities eliminate the need for domain-specific action representations, while pressure-field coordination eliminates the need for complex multi-agent protocols, together enabling simple yet effective multi-agent systems.

These results suggest that for domains with measurable quality signals, implicit coordination through shared state offers a simpler, equally effective alternative to explicit hierarchical control—and a strictly superior alternative to dialogue-based coordination.

9 Appendix: Experimental Protocol

This appendix provides complete reproducibility information for all experiments.

9.1 Hardware and Software

Hardware: NVIDIA A100 80GB GPU (RunPod cloud)

Software:

- Rust 1.75+ (edition 2024)
- vLLM (OpenAI-compatible inference server)
- Models: `Qwen/Qwen2.5-0.5B`, `Qwen/Qwen2.5-1.5B`, `Qwen/Qwen2.5-3B`, `Qwen/Qwen2.5-7B`, `Qwen/Qwen2.5-14B`

9.2 Model Configuration

Models are served via vLLM with a system prompt configured for Latin Square solving:

```
You solve Latin Square puzzles. Given a row with empty cells (_),
return ONLY the number(s) that fill them. Return just the numbers,
nothing else.
```

For multi-model setups (model escalation), each model runs on a separate vLLM instance with automatic port routing based on model size.

9.3 Sampling Diversity

The experiment framework overrides default sampling parameters with three exploration bands per LLM call:

Band	Temperature	Top-p
Exploitation	0.15 - 0.35	0.80 - 0.90
Balanced	0.35 - 0.55	0.85 - 0.95
Exploration	0.55 - 0.85	0.90 - 0.98

Table 8: Sampling parameter ranges. Each LLM call randomly samples from one band.

This diversity prevents convergence to local optima and enables exploration of the solution space.

9.4 Experiment Commands

Main Grid (Strategy Comparison):

```
latin-experiment --vllm-host http://localhost:8001 \
  --model-chain "Qwen/Qwen2.5-0.5B,Qwen/Qwen2.5-1.5B,Qwen/Qwen2.5-3B,Qwen/
Qwen2.5-7B,Qwen/Qwen2.5-14B" \
  --escalation-threshold 20 \
  grid --trials 30 --n 7 --empty 7 --max-ticks 100 --agents 1,2,4,8
```

Ablation Study:

```
latin-experiment --vllm-host http://localhost:8001 \
  --model-chain "Qwen/Qwen2.5-0.5B" \
  ablation --trials 30 --n 7 --empty 7 --max-ticks 100
```

Scaling Analysis:

```
latin-experiment --vllm-host http://localhost:8001 \
  --model-chain "Qwen/Qwen2.5-0.5B,Qwen/Qwen2.5-1.5B,Qwen/Qwen2.5-3B,Qwen/
Qwen2.5-7B,Qwen/Qwen2.5-14B" \
  --escalation-threshold 20 \
  grid --trials 30 --n 7 --empty 8 --max-ticks 100 --agents 1,2,4,8,16,32
```

Model Escalation Comparison:

```
# Without escalation (single model)
latin-experiment --vllm-host http://localhost:8001 \
  --model-chain "Qwen/Qwen2.5-0.5B" \
  grid --trials 30 --n 7 --empty 8 --max-ticks 100 --agents 2,4,8

# With escalation (full chain)
latin-experiment --vllm-host http://localhost:8001 \
  --model-chain "Qwen/Qwen2.5-0.5B,Qwen/Qwen2.5-1.5B,Qwen/Qwen2.5-3B,Qwen/
Qwen2.5-7B,Qwen/Qwen2.5-14B" \
  --escalation-threshold 20 \
  grid --trials 30 --n 7 --empty 8 --max-ticks 100 --agents 2,4,8
```

Difficulty Scaling:

```
# Easy (5x5, 5 empty)
latin-experiment --vllm-host http://localhost:8001 \
  --model-chain "Qwen/Qwen2.5-0.5B,Qwen/Qwen2.5-1.5B,Qwen/Qwen2.5-3B,Qwen/
Qwen2.5-7B,Qwen/Qwen2.5-14B" \
  --escalation-threshold 20 \
  grid --trials 30 --n 5 --empty 5 --max-ticks 100 --agents 4
```

9.5 Metrics Collected

Each experiment records:
- `solved`: Boolean indicating puzzle completion
- `total_ticks`: Iterations to solve (or max if unsolved)
- `pressure_history`: Pressure value at each tick
- `escalation_events`: Model tier changes (tick, from_model, to_model)
- `final_model`: Which model tier solved the puzzle

9.6 Replication Notes

Each configuration runs 30 independent trials with different random seeds to ensure reliability. Results report mean solve rates and tick counts across trials.

9.7 Estimated Runtime

Experiment	Configurations	Trials	Est. Time
Main Grid	20	30	2 hours
Ablation	8	30	1 hour
Scaling	30	30	3 hours
Escalation	10	30	2 hours
Difficulty	5	30	1.5 hours
Total			**9.5 hours**

Table 9: Estimated runtime for all experiments on NVIDIA A100 80GB GPU with 10 parallel jobs.

10 Appendix B: Conversation Protocol

This appendix provides the complete protocol for the Conversation baseline strategy, demonstrating that it faithfully implements AutoGen-style multi-agent dialogue coordination.

10.1 Agent Roles

The Conversation strategy employs three specialized agents, each with distinct responsibilities:

Coordinator Agent: Observes the full puzzle state and selects which region (row) to target. After the Proposer/Validator dialogue, synthesizes the final decision (APPLY or REJECT).

Proposer Agent: Given a target region and column availability constraints, proposes a single value for one empty cell. Has access to the conversation history to avoid repeating rejected proposals.

Validator Agent: Critiques proposals against Latin Square constraints. Checks for row duplicates, column conflicts, and range violations. Outputs APPROVE or REJECT with reason.

10.2 Protocol Pseudocode

```
CONVERSATION_TICK(artifact, shared_grid):
  state ← new ConversationState(max_turns=5)

  // TURN 1: Coordinator selects target region
  puzzle_state ← format_puzzle(artifact)
  prompt ← COORDINATOR_SELECT_TEMPLATE(puzzle_state)
  response ← LLM(prompt)
  region_id ← parse_target_row(response)
  state.add_message(COORDINATOR, response)

  // TURNS 2-N: Proposer/Validator dialogue
  last_approved ← false
  FOR turn IN 1..max_turns:
    // Proposer turn
    availability ← get_column_availability(artifact, region_id)
    prompt ← PROPOSER_TEMPLATE(region_content, availability, state.history)
    response ← LLM(prompt)
```

```
    (position, value) ← parse_proposal(response)
    state.add_message(PROPOSER, response)

    IF proposal_valid:
      // Validator turn
      col_values ← get_column_values(shared_grid, position)
      row_values ← get_row_values(region_content)
      prompt ← VALIDATOR_TEMPLATE(region_content, proposal, col_values,
row_values)
      response ← LLM(prompt)
      state.add_message(VALIDATOR, response)

      IF response contains 'APPROVE':
        patch ← construct_patch(region_content, position, value)
        RETURN (patch, state)

  // No consensus reached
  RETURN (None, state)
```

10.3 Prompt Templates

Each agent receives a structured prompt designed to elicit the expected behavior:

Coordinator Selection Prompt:

```
You are a Coordinator agent solving a {n}x{n} Latin Square puzzle.
Current puzzle state (each row is numbered, _ means empty):
{puzzle_state}

Task: Identify which row needs the most attention. Consider:
1. Rows with empty cells
2. Rows with constraint violations

Respond with ONLY: TARGET row=<N>
```

Proposer Prompt:

```
You are a Proposer agent solving a Latin Square puzzle.
Target row {row_idx}: {region_content}
Available values per column position: {availability}
Previous messages: {history}

Propose ONE value for ONE empty cell (_).
Format: PROPOSE position=<col> value=<num>
```

Validator Prompt:

```
You are a Validator agent checking Latin Square constraints.
Row: {region_content}
Proposal: {proposal}
Values already in target column: {column_values}
Values already in row: {row_values}

Check if the proposed value violates constraints.
Respond with ONLY: APPROVE or REJECT <reason>
```

10.4 Key Design Decisions

1. **Sequential Message Passing:** A semaphore enforces that only one LLM call executes at a time within a conversation, mimicking AutoGen's turn-based dialogue.

2. **Same LLM as Other Strategies:** All agents use the same model (Qwen2.5 series with escalation), ensuring the comparison isolates coordination mechanism effects.
3. **Same Patch Validation:** Successful proposals undergo identical validation as other strategies—patches that increase violations are rejected.
4. **Explicit Consensus Requirement:** Unlike pressure-field where any pressure-reducing patch is accepted, Conversation requires explicit Validator approval.

10.5 Overhead Analysis

Each Conversation tick requires $3 + 2 \cdot (\text{turns} - 1)$ LLM calls in the worst case (Coordinator select + N rounds of Proposer/Validator). With max_turns=5, this is up to 11 sequential LLM calls per tick versus 1 parallel batch for pressure-field. This sequential overhead contributes to the strategy's poor performance—the coordination cost dominates any potential benefit from explicit negotiation.

Bibliography

[1] Q. Wu *et al.*, "AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation," *arXiv preprint arXiv:2308.08155*, 2023.

[2] S. Hong *et al.*, "MetaGPT: Meta Programming for A Multi-Agent Collaborative Framework," *arXiv preprint arXiv:2308.00352*, 2023.

[3] G. Li, H. A. A. K. Hammoud, H. Itani, D. Khizbullin, and B. Ghanem, "CAMEL: Communicative Agents for "Mind" Exploration of Large Language Model Society," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[4] B. Horling and V. Lesser, "A Survey of Multi-agent Organizational Paradigms," *The Knowledge Engineering Review*, vol. 19, no. 4, pp. 281–316, 2004.

[5] T. W. Malone and K. Crowston, "The Interdisciplinary Study of Coordination," *ACM Computing Surveys*, vol. 26, no. 1, pp. 87–119, 1994.

[6] K. Decker and V. Lesser, "Designing a Family of Coordination Algorithms," in *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, 1995, pp. 73–80.

[7] B. J. Grosz and S. Kraus, "Collaborative Plans for Complex Group Action," *Artificial Intelligence*, vol. 86, no. 2, pp. 269–357, 1996.

[8] T. De Wolf and T. Holvoet, "Towards a Methodology for Engineering Self-Organising Emergent Systems," *Self-Organization and Autonomic Informatics (I)*. IOS Press, 2005.

[9] CrewAI, "Framework for orchestrating role-playing, autonomous AI agents." 2024.

[10] P.-P. Grassé, "La reconstruction du nid et les coordinations interindividuelles chez Bellicositermes natalensis et Cubitermes sp. La théorie de la stigmergie," *Insectes Sociaux*, vol. 6, pp. 41–80, 1959.

[11] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant System: Optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man, and Cybernetics – Part B*, vol. 26, no. 1, pp. 29–41, 1996.

[12] M. Dorigo and L. M. Gambardella, "Ant Colony System: A cooperative learning approach to the traveling salesman problem," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 53–66, 1997.

[13] D. Monderer and L. S. Shapley, "Potential Games," *Games and Economic Behavior*, vol. 14, pp. 124–143, 1996.

[14] A. Nedić and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Transactions on Automatic Control*, vol. 54, no. 1, pp. 48–61, 2009.

[15] K. Yuan, Q. Ling, and W. Yin, "On the convergence of decentralized gradient descent," *SIAM Journal on Optimization*, vol. 26, no. 3, pp. 1835–1854, 2016.

[16] Y. Shoham and K. Leyton-Brown, *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge University Press, 2008.