THE EVOLUTION OF MIND: EMERGENCE OF COLLECTIVE INTELLIGENCE THROUGH LOGICAL-PROBABILISTIC KNOWLEDGE DYNAMICS IN MULTI-AGENT ENIGMA METAVERSE ECOSYSTEMS

Andrey Nechesov[1], Sergey Barykin[1] and Janne Ruponen[1]

[1] *International Artificial Intelligence Committee (IAIC)*

nechesoff@gmail.com, sbe@list.ru, ruponez@gmail.com

ABSTRACT

Modern metaverse platforms, populated by heterogeneous multi-agent systems (MAS), generate vast streams of experiential data whose epistemic value remains largely untapped. This paper introduces the **Enigma framework**—a formal theory of *collective intelligence emergence* in metaverse ecosystems, grounded in a novel **logical-probabilistic learning theory** that extends classical first-order logic with probabilistic confidence annotations and distributed knowledge semantics. We define a *Distributed Knowledge Lattice* (DKL) over multi-agent interactions [33] and prove that, under precisely stated monotonicity and convergence conditions, the collective knowledge of an agent population forms a complete lattice whose least upper bound represents an emergent cognitive state unreachable by any individual agent. We formalize the dynamics of knowledge creation, verification, and propagation through a system of *Logical-Probabilistic Agents* (LP-agents) that interact with LLM-driven entities, providing trustworthy and explainable reasoning via symbolic proof chains stored on a multi-blockchain ledger. Central results include: (i) a **Collective Intelligence Convergence Theorem** establishing conditions under which the system's aggregate knowledge monotonically approaches a fixed point; (ii) a **Probabilistic Inference Soundness Theorem** guaranteeing that confidence propagation through distributed reasoning chains preserves logical consistency; and (iii) a polynomial-time algorithm for optimal knowledge retrieval from the distributed lattice. The framework is instantiated within the Enigma Metaverse architecture, where smart contracts govern knowledge tokenization, cross-chain knowledge interoperability protocols enable seamless sharing, and decentralized governance mechanisms ensure epistemic accountability. We demonstrate that this synthesis of mathematical logic, probability theory, LLM-based hypothesis generation, and blockchain-secured knowledge persistence provides a rigorous foundation for building self-optimizing, trustworthy, and explainable collective intelligence (CI) in virtual worlds.

Keywords: Collective Intelligence, Metaverse, Multi-Agent Systems, Logical-Probabilistic Learning Theory, Distributed Knowledge Lattice, Explainable AI, Trustworthy AI, Large Language Models, Blockchain, Smart Contracts

1 INTRODUCTION

The pursuit of artificial intelligence systems that exhibit genuine collective cognition—intelligence that transcends the sum of its individual agents—represents one of the deepest open problems at the intersection of mathematical logic, probability theory, and distributed computing [16, 21]. While contemporary Large Language Models (LLMs) demonstrate impressive individual reasoning capabilities [1, 20], they remain fundamentally isolated: each model instance operates within its own ephemeral context window, lacking mechanisms for cumulative, verifiable knowledge sharing with other intelligent entities.

The metaverse—conceived as a persistent, immersive virtual universe populated by autonomous agents [24]—offers a unique substrate for studying and engineering collective intelligence [10, 12]. Unlike static benchmarks, a metaverse environment demands that agents continuously learn, adapt, cooperate, and compete within a shared world governed by formal rules. This paper argues that the metaverse is not merely a testbed for AI but an *active cognitive ecosystem* where collective intelligence (CI) naturally emerges from the structured interactions of heterogeneous agents.

Received: March 15, 2026 Accepted: April 1, 2026 Published: April 4, 2026

The central challenge we address is: *Under what mathematical conditions does a population of interacting agents, each possessing partial and probabilistic knowledge, give rise to a coherent CI that is provably more capable than any individual?* Answering this requires a formal framework that unifies three traditionally separate domains:

(i) **Mathematical Logic**: providing the inferential backbone for rigorous, verifiable reasoning;

(ii) **Probability Theory**: accounting for the inherent uncertainty in agent-generated knowledge;

(iii) **Distributed Systems**: ensuring that collective knowledge is persistent, tamper-proof, and accessible.

Our approach builds upon and substantially extends the task-based cognitive architecture and the logical-probabilistic learning theory introduced for virtual cities [12], while proposing a fundamentally new mathematical object—the **Distributed Knowledge Lattice** (DKL)—as the formal substrate for CI.

1.1 Contributions

The principal contributions of this paper are:

1. **A Formal Theory of CI Emergence**: We define CI as a fixed-point property of a knowledge lattice and prove convergence under realistic conditions (Section 3).
2. **Logical-Probabilistic Knowledge Dynamics**: We introduce a dynamical system governing the evolution of probabilistic knowledge across a multi-agent population, with soundness guarantees for distributed inference (Section 4).
3. **LP-Agent Verification Framework**: We formalize the interaction between LLM-agents and Logical-Probabilistic agents, providing explainability [26] and trustworthiness [25] guarantees (Section 5).
4. **The Enigma Architecture**: We instantiate the theory within a concrete multi-blockchain metaverse platform with smart-contract-governed knowledge economics (Section 6).
5. **Convergence Analysis and Complexity Results**: We provide precise convergence rates and prove that optimal knowledge retrieval from the DKL is polynomial (Section 7).

2 Related Work

Neuro-Symbolic AI and LLM Verification. The integration of neural generation with symbolic verification has gained significant traction. Logical Neural Networks [15, 23] embed propositional logic directly into network architectures. Logic-LM [13] and PAL [4] use external symbolic solvers to verify LLM outputs. The TBCA framework established a closed cognitive loop where LLMs generate hypotheses verified by a symbolic engine, with results stored on blockchain. Our work extends this by formalizing the *inter-agent* dynamics of knowledge creation and proving emergent properties of the collective.

Multi-Agent Systems and CI. Classical MAS theory [17, 21] addresses coordination, negotiation, and emergent behavior but typically lacks formal treatments of knowledge dynamics. Recent work on generative agents [32] and CAMEL [11] demonstrates emergent social behaviors in LLM-based agent societies. Project Sid [22] simulates civilizations with 1000 agents. However, these works lack formal guarantees on knowledge quality, convergence, or trustworthiness.

Blockchain-Based Knowledge Management. The axiomatization of blockchain theory [6] provides formal foundations for distributed ledger properties. Applications to AI knowledge management [9] demonstrate tamper-proof knowledge sharing. Our work introduces *knowledge tokenization* as an economic mechanism for incentivizing CI growth.

Metaverse as AI Research Platform. Virtual cities [12] and digital twins [7, 18] provide immersive environments for AI testing. The concept of Artificial Collective Consciousness (ACC) [12] and DAO-based governance [2] inform our approach to epistemic governance in the Enigma ecosystem.

Probabilistic Logic and Learning Theory. Probabilistic logic programming [3] and statistical relational learning [5] combine logical structure with probabilistic inference. Our logical-probabilistic learning theory draws from the task-based approach [12] but extends it with lattice-theoretic semantics and distributed consensus properties.

The Openness Problem. Multi-agent systems in open environments face three types of dynamism: 1) Agents join (onboarding) and leave continuously (exit and slashing), violating the fixed set assumption of traditional MARL; 2) Transactions appear (submission) and disappear (replacement, reorganization) dynamically, breaking the stable reward function assumption; and 3) evolution of validator capability (upgrades), preference shift (MEV strategies), and goal adjustment (honest to Byzantine) invalidating mapping the coherent action-outcome dynamics. Studies show that openness degrades MARL performance by 40% for single openness types, while catastrophic collapse under combined openness [27]. This quantifies the cost of decentralization and motivates architectural solutions.

Credit Assignment Problem. Credit assignment attributes shared rewards to individual agent contributions, and is identified as the multi-agent coordination problem [28, 29, 31]. This problem is divided into temporal credit assignment (TCA) and structural credit assignment (SCA) [27]. TCA addresses delayed rewards by attributing a reward received at time t to actions taken a priori $t - k$. In blockchain, TCA links validator actions to future block rewards despite epoch delays. SCA addresses multi-agent attribution by distributing a shared team reward among individual contributions by agents.

Traditional approaches (VDN, QMIX) fail under openness due of assumption of stable team structure [27]. Polarization policy gradients (MAPPG) solve this by amplifying the reward gap between optimal and non-optimal behaviors [29] via memory substrates that can track agent contributions over time.

Persistent Memory in Multi-Agent Coordination Recent research demonstrate that cognitive CI in MAS emerges from persistent agent interactions when equipped with proper memory architectures [32, 35]. However, typical MAS implementations reset memory after each session, preventing the formation of long-term relationships, knowledge accumulation, and adaptive coordination strategies that define intelligent systems.

While MADRL focuses on training paradigms and reward-driven behaviors, LLM-based MAS adds distinct challenges centered on memory management and contextual reasoning. Memory types have been divided in earlier studies to five categories: short-term memory for live interactions, long-term memory storing historical queries, external data storage for knowledge augmentation, episodic memory capturing past events of multi-agent interactions, and consensus memory providing distributed knowledge across agents [30]. These memory types operate within complex hierarchical access control, and are required to maintain consistency across overlapping agent knowledge while enabling effective retrieval of past interactions based on contextual relevance.

Memory is essential for agents to recall prior subtasks, coordinate handoffs, maintain role histories, and build on earlier steps rather than starting from scratch [30], yet virtually all current MAS implementations treat memory in session-limited scope, in which agents reset each interaction. There is no architecture in the literature providing a hardware-efficient, distributed, always-on memory substrate at the agent level. Management of memory storage has been identified as playing a critical role in collaboration between agents, providing constraints to context alignment, adaptive learning and associative recall [30].

Attempts to establish persistent agent memory in simulated social environments have relied on recency, importance, and reflection as retrieval signals [32]. However, this strategy increases computational costs as each memory query requires scoring the entire history with complex weighting functions, while no providing mechanisms for forgetting obsolete information or representing uncertainty.

Game theory combined with blockchain networks constitutes partially observable stochastic games in which validators must coordinate under uncertainty [31]. The non-stationarity inherent in open blockchain systems violates the assumptions of traditional reinforcement learning [27], requesting novel memory architectures for dynamic, unbounded agent populations.

MARL Problems. Multi-agent reinforcement learning (MARL) surveys have identified existing challenges stemming from credit assignment problem (CAP), non-stationary interactions, and scalability due to exponentially growing state-action space [34]. MARL has undergone a rapid transformation since the introduction of deep learning (DL) methods, enabling agents to act on real-world complexity that was previously unmanageable with traditional tabular approaches. In multi-agent deep reinforcement learning (MADRL), autonomous agents interact simultaneously within a shared environment while learning policies through trial-and-error and adapting to the behavioral changes of other agents. The one of the applicable frameworks for MADRL is the Markov Game, which extends the single-agent Markov Decision Process (MDP) to multiple decision-makers, each with individual action spaces and reward functions [34].

MADRL depends on the training paradigm employed to learn agent policies. Gronauer and Diepold identify three primary approaches: distributed training with decentralized execution (DTDE), where agents learn independently;

centralized training with centralized execution (CTCE), where a single joint policy governs all agents; and centralized training with decentralized execution (CTDE), which has emerged as the state-of-the-art approach [34]. The CTDE paradigm allows agents to share information such as events, functions and policy parameters during training while acting independently at deployment. This addresses the challenge of non-stationarity, where the environment appears to change from each agent's perspective as others simultaneously adjust their own policies.

The CTDE paradigm resolves the centralized-decentralized mismatch problem, where weak agent policies corrupt reward attribution for others. However, its applicability is limited in open blockchain environments where validator sets rotate continuously as CTDE assumes stable team structure during training.

Non-Stationarity and Experience Replay Obsolescence The non-stationarity problem in MARL emerge as each agent's environment includes the policies of other agents, each evolving individually during learning [31]. The state transition function from agent's perspective becomes non-stationary. Meanwhile, typical experience replay stores transitions and samples them uniformly during training. However, when contrasting policies evolve, stored experiences return outdated strategies. For example, a validator's learned response to a transaction submission pattern becomes obsolete when other validators adopt new MEV extraction strategies. Replaying obsolete experiences can also lead to policy degradation, as the agent reinforces behaviors that were optimal under previous opponent policies but are now flawed [27].

Recent approaches attempt to mitigate this through techniques such as importance sampling with behavioral cloning, fingerprinting opponent policies, and meta-learning for rapid adaptation [31]. However, these methods require either explicitly tracking opponent policy parameters (violating privacy in decentralized settings) or maintaining multiple policy snapshots (increasing memory overhead linearly with history length).

In blockchain-based validator networks, non-stationarity manifests in three stages: (1) validator set rotation (agent openness), where new validators join with unknown strategies; (2) protocol upgrades (type openness), where consensus rules change mid-operation; and (3) evolving MEV strategies (task openness), where transaction ordering games shift continuously.

Consequently, a non-stationary memory system must handle three conditions. Decay obsolete experiences need to automatically reduce the influence of outdated strategic interactions without explicit forgetting logic. Preserve temporal context maintain the time-ordering of experiences to distinguish recent (relevant) from distant (obsolete) patterns. Rapid adaptation supports quick retrieval of similar past situations when environment shifts occur.

Vector databases with uniform retrieval probability fail to satisfy these requirements, as they treat all stored vectors equally regardless of age. Graph databases are able to model temporal relationships explicitly, but require also expensive traversal queries to determine experience recency. The more efficient database models should therefore seek to apply intrinsic decay mechanisms, where memory activation strengths diminish naturally over time without reinforcement. This approach would seek to mitigate computational overhead by providing automated recency weights.

3 Mathematical Foundations: Logical-Probabilistic Knowledge Theory

3.1 Probabilistic Knowledge Units

We begin by formalizing the fundamental unit of knowledge in our framework.

Definition 3.1 (Probabilistic Knowledge Unit)**.** A *probabilistic knowledge unit* (PKU) is a triple

$$K = \langle \forall \bar{x}\, \exists \bar{y}\, (\Phi(\bar{x}, \bar{y}) \to \Psi(\bar{x}, \bar{y})),\ \bar{y} = t(\bar{x}),\ p \rangle$$

where Φ, Ψ are well-formed formulas in first-order logic, $\bar{y} = t(\bar{x})$ is a computable solution term, and $p \in [0, 1]$ is the confidence score representing the probability that the solution is correct.

A *fact* is a knowledge unit with $p = 1$ and fully instantiated variables. A knowledge unit with $p < 1$ is called *probabilistic* or *a posteriori*.

Definition 3.2 (Knowledge Ordering)**.** We define a partial order $\preceq$ on the set of all knowledge units: $K_1 \preceq K_2$ if and only if:

(a) the premise Φ_1 of K_1 is logically implied by the premise Φ_2 of K_2,

(b) the conclusion Ψ_1 of K_1 is logically implied by Ψ_2, and

(c) the confidence satisfies $p_1 \leq p_2$.

PKU Deployment. The PKU structure (Definition 3.1) is realized in both TypeScript and Rust with simplified representations pending full first-order logic integration. The current implementation uses string-based premise/conclusion pairs extracted via regex patterns, with solution terms represented as literal strings rather than formal λ-calculus terms. Confidence scores are stored as basis points (0–10000) for precision in smart contract arithmetic. The `to_formula()` method serializes PKUs for blockchain storage in the format `"IF {premise} THEN {conclusion} [solution: {solution}, confidence: {p}]"`, enabling human-readable audit trails. Content-addressable identifiers via `keccak256` enable efficient deduplication across the distributed system. A seed knowledge base of PKUs is generated using the multi-LLM system, demonstrating the LLM hypothesis generation pipeline. Full integration of formal logical formula parsing and proof chain construction is planned for Phase 2 development.

3.2 The Distributed Knowledge Lattice

We now introduce the central mathematical structure of this paper.

Definition 3.3 (Agent Knowledge Base)**.** For an agent A_i in a population $\mathcal{A} = \{A_1, \ldots, A_n\}$, the *knowledge base* $\mathcal{K}_i$ is a finite set of probabilistic knowledge units closed under the inference rules of Section 3.3.

Definition 3.4 (Distributed Knowledge Lattice)**.** The *Distributed Knowledge Lattice* (DKL) of an agent population $\mathcal{A}$ is the structure

$$\mathcal{L}(\mathcal{A}) = \left(2^{\bigcup_{i=1}^{n} \mathcal{K}_i},\ \preceq,\ \sqcap,\ \sqcup\right)$$

where:

- $\sqcap$ (meet) is defined by $S_1 \sqcap S_2 = \{K \in S_1 \cap S_2 \mid K \text{ is consistent}\}$;
- $\sqcup$ (join) is defined by $S_1 \sqcup S_2 = \mathrm{Cl}(S_1 \cup S_2)$, where $\mathrm{Cl}(\cdot)$ denotes deductive closure under the probabilistic inference rules with consistency checking;
- the bottom element $\bot = \emptyset$;
- the top element $\top = \mathrm{Cl}\left(\bigcup_{i=1}^{n} \mathcal{K}_i\right)$, the maximal consistent closure of all agent knowledge.

Theorem 3.5 (Lattice Completeness)**.** *$\mathcal{L}(\mathcal{A})$ forms a complete lattice under the ordering induced by $\preceq$ on knowledge sets.*

Proof. We must show that every subset $\mathcal{S} \subseteq \mathcal{L}(\mathcal{A})$ has both a greatest lower bound (infimum) and a least upper bound (supremum). Since $\mathcal{L}(\mathcal{A})$ is a power set ordered by the extended $\preceq$, and the closure operator $\mathrm{Cl}(\cdot)$ is monotone and idempotent on finite sets of knowledge units (monotonicity follows from the confidence monotonicity property—adding consistent knowledge cannot decrease the confidence of existing derivations), the structure satisfies the conditions of the Knaster–Tarski theorem [19]. The infimum of $\mathcal{S}$ is $\bigsqcap \mathcal{S} = \bigcap_{S \in \mathcal{S}} S$ (restricted to consistent units), and the supremum is $\bigsqcup \mathcal{S} = \mathrm{Cl}\left(\bigcup_{S \in \mathcal{S}} S\right)$. Finiteness of agent knowledge bases ensures well-definedness. □

DKL Implementation Status. The DKL (Definitions 3.3–3.4, Theorem 3.5) provides the formal mathematical substrate for CI emergence. The lattice operations (meet $\sqcap$, join $\sqcup$, closure $\mathrm{Cl}(\cdot)$, ordering $\preceq$) are fully implemented in `crates/knowledge/src/lattice.rs` and `inference.rs`. The `meet()` function implements intersection with consistency checking, `join()` computes union with deductive closure, and `closure()` applies inference rules iteratively until reaching a fixed point. The `bottom()` and `top()` elements are constructors for $\bot$ and $\top$ respectively. Individual agent knowledge bases are maintained as HashSets of PKUs with Eq and Hash trait implementations for efficient lattice operations, with consistency checking implemented through content-hash deduplication.

A real-time interactive visualization of the DKL has been deployed in the SvelteKit frontend using D3.js force-directed graphs, showing PKU nodes with color-coded layers (Stream=yellow, Logic=green, Anchor=blue), inference edges (green dashed lines), and subsumption edges (blue solid lines). The visualization demonstrates the lattice structure with $\bot$ (bottom) connected to base PKUs and $\top$ (top) representing maximal consistent closure. Full lattice structure computation—including supremum/infimum over arbitrary knowledge subsets, topological ordering under $\preceq$, and fixed-point detection per Theorem 4.4—requires a populated multi-agent knowledge base with sufficient density to demonstrate non-trivial deductive closure properties. The architectural design anticipates DKL materialization as an in-memory index over Layer 2 blockchain state, enabling efficient lattice queries during agent reasoning cycles.

3.3 Inference Rules for Probabilistic Knowledge

The following rules govern knowledge derivation within the lattice.

Definition 3.6 (Probabilistic Modus Ponens)**.** Given knowledge units $K_1 = \langle A, t_1, p_1 \rangle$ and $K_2 = \langle A \to B, t_2, p_2 \rangle$, we derive

$$K_3 = \langle B, t_2 \circ t_1, p_1 \cdot p_2 \rangle$$

The derived confidence $p_3 = p_1 \cdot p_2$ reflects the multiplicative uncertainty accumulation.

Definition 3.7 (Confidence Threshold)**.** A derived knowledge unit K is *accepted* into the knowledge base if $\mathrm{conf}(K) \geq \theta$, where $\theta \in (0, 1)$ is a system-wide acceptance threshold.

Remark 3.8 (Non-Transitivity of Probabilistic Inference)**.** A critical observation motivating our framework: given $K_1 : A \to B$ with confidence p_1 and $K_2 : B \to C$ with confidence p_2, even if p_1, p_2 are close to 1, the derived knowledge $K_3 : A \to C$ has confidence $p_1 \cdot p_2$, which may fall below threshold θ for long inference chains. This **confidence decay** property fundamentally distinguishes probabilistic from classical logical inference and necessitates the enrichment mechanisms described in Section 4.

Definition 3.9 (Statistical Generalization)**.** Given k positive instances and $n - k$ negative instances of a formula $F(\bar{x}, \bar{y})$, we derive:

$$K_{\text{gen}} = \left\langle F(\bar{x}, \bar{y}),\ \bar{y} = t(\bar{x}),\ \frac{k}{n} \right\rangle$$

Definition 3.10 (Knowledge Enrichment Operator)**.** The *enrichment operator* $\mathcal{E} : 2^{\mathcal{K}} \times 2^{\mathcal{K}} \to 2^{\mathcal{K}}$ is defined as:

$$\mathcal{E}(\mathcal{K}_i, \Delta\mathcal{K}) = \mathrm{Cl}(\mathcal{K}_i \cup \{K \in \Delta\mathcal{K} \mid \mathrm{conf}(K) \geq \theta \text{ and } K \text{ is consistent with } \mathcal{K}_i\})$$

Theorem 3.11 (Confidence Monotonicity under Enrichment)**.** *Let h be a hypothesis, $\mathcal{K}$ a knowledge base, and $\mathcal{K}' = \mathcal{E}(\mathcal{K}, \Delta\mathcal{K})$. Then:*

$$\mathrm{conf}(h \mid \mathcal{K}') \geq \mathrm{conf}(h \mid \mathcal{K})$$

Proof. The confidence of h under context $\mathcal{K}$ is computed as $\mathrm{conf}(h \mid \mathcal{K}) = \max_{\pi \in \Pi(h,\mathcal{K})} \prod_{j=1}^{|\pi|} \alpha_{i_j}$, where $\Pi(h, \mathcal{K})$ is the set of all valid derivation chains for h using rules in $\mathcal{K}$. Since $\mathcal{K} \subseteq \mathcal{K}'$, every derivation in $\Pi(h, \mathcal{K})$ is also in $\Pi(h, \mathcal{K}')$, so $\Pi(h, \mathcal{K}) \subseteq \Pi(h, \mathcal{K}')$. Additionally, the enriched knowledge $\Delta\mathcal{K}$ may enable new derivation paths with higher confidence (by filling gaps, shortening chains, or providing alternative proofs). Since confidence is the maximum over all valid derivations, the result follows. □

Inference Rules Implementation Status. Probabilistic modus ponens (Definition 3.6) is implemented in `inference.rs` with multiplicative confidence propagation: given $K_1 = \langle A, t_1, p_1 \rangle$ and $K_2 = \langle A \to B, t_2, p_2 \rangle$, the system derives $K_3 = \langle B, t_2 \circ t_1, p_1 \cdot p_2 \rangle$. The enrichment operator $\mathcal{E}$ (Definition 3.10) filters incoming knowledge by confidence threshold ($\theta = 0.7$ default) and consistency checking before applying deductive closure. Statistical generalization is implemented for deriving generalized rules from multiple instances. The confidence threshold filter gates admission to Layer 2 blockchain storage. Full activation of the enrichment operator awaits population of a dense knowledge base with sufficient cross-agent knowledge exchange patterns.

4 CI Dynamics

4.1 The Knowledge Evolution System

We model the temporal evolution of collective knowledge as a discrete dynamical system over the DKL.

Definition 4.1 (Knowledge State)**.** At time step $t \in \mathbb{N}$, the *collective knowledge state* is:

$$\Sigma(t) = \bigsqcup_{i=1}^{|\mathcal{A}|} \mathcal{K}_i(t)$$

where $\mathcal{K}_i(t)$ is the knowledge base of agent A_i at time t.

Definition 4.2 (Knowledge Update Rule)**.** Each agent A_i updates its knowledge base at each time step via:

$$\mathcal{K}_i(t+1) = \mathcal{E}\left(\mathcal{K}_i(t),\ \bigcup_{j \in \mathcal{N}(i)} \Delta\mathcal{K}_j(t)\right) \tag{1}$$

where $\mathcal{N}(i) \subseteq \mathcal{A}$ is the communication neighborhood of A_i and $\Delta\mathcal{K}_j(t)$ is the set of newly validated knowledge units produced by agent A_j at time t.

Definition 4.3 (CI Measure)**.** The *CI* of the system at time t is quantified by:

$$\mathrm{CI}(t) = \frac{|\Sigma(t)|}{|\Sigma^*|} \cdot \bar{p}(t) \cdot \left(1 + \frac{C_s(t)}{T_s(t)}\right) \tag{2}$$

where $|\Sigma(t)|$ is the cardinality of the collective knowledge state, $|\Sigma^*|$ is the theoretical maximum (cardinality of $\top$ in the DKL), $\bar{p}(t) = \frac{1}{|\Sigma(t)|}\sum_{K\in\Sigma(t)} \mathrm{conf}(K)$ is the mean confidence, $C_s(t)$ is the count of collaboratively solved problems, and $T_s(t)$ is the total attempted.

4.2 The CI Convergence Theorem

We now state and prove the central result of this paper.

Theorem 4.4 (CI Convergence)**.** *Let $\mathcal{A}$ be a finite population of agents with pairwise-connected communication graph, operating under the update rule equation 1 with confidence threshold $\theta > 0$. Assume:*

- *(C1)* ***Finite Knowledge Domain****: The set of all possible knowledge units is finite with cardinality N^*.*
- *(C2)* ***Consistency Preservation****: The enrichment operator $\mathcal{E}$ preserves logical consistency.*
- *(C3)* ***Non-Trivial Generation****: At each step, at least one agent produces at least one new validated knowledge unit with confidence $\geq \theta$, until no further derivations are possible.*

Then there exists a finite time $T^* \leq N^* \cdot \mathrm{diam}(\mathcal{N})$ such that:

$$\Sigma(t) = \Sigma(T^*) = \Sigma^{\mathrm{fix}} \quad \textit{for all } t \geq T^*$$

where Σ^{fix} is the unique least fixed point of the collective update operator, and:

$$\mathrm{CI}(T^*) \geq \mathrm{CI}(t) \quad \textit{for all } t < T^*$$

Proof. The proof proceeds in three steps.

Step 1: Monotonicity. By Theorem 3.11, the enrichment operator is monotone: $\mathcal{K}_i(t) \subseteq \mathcal{K}_i(t+1)$ for all i, t (set inclusion up to consistency). Therefore $\Sigma(t) \sqsubseteq \Sigma(t+1)$ in the DKL ordering.

Step 2: Boundedness. By condition (C1), $|\Sigma(t)| \leq N^*$ for all t. The sequence $\{|\Sigma(t)|\}_{t\geq 0}$ is monotonically non-decreasing and bounded above.

Step 3: Termination. By conditions (C1) and (C3), each time step either increases $|\Sigma(t)|$ or the system has reached a fixed point. Since the knowledge domain is finite, the system must reach a fixed point in at most N^* steps. The communication diameter $\mathrm{diam}(\mathcal{N})$ accounts for the propagation delay across the agent network, giving the bound $T^* \leq N^* \cdot \mathrm{diam}(\mathcal{N})$.

The monotonicity of $\mathrm{CI}(t)$ follows from the monotonicity of each factor in equation 2: $|\Sigma(t)|$ is non-decreasing, $\bar{p}(t)$ is non-decreasing by the threshold filter (only units with $\mathrm{conf} \geq \theta$ are admitted), and $C_s(t)/T_s(t)$ is non-decreasing as collaborative problem solving accumulates validated solutions. □

4.3 Convergence Rate Analysis

Proposition 4.5 (Exponential Knowledge Growth Phase)**.** *Under conditions (C1)–(C3), if the communication graph $\mathcal{N}$ is an expander with spectral gap $\lambda > 0$, then during the growth phase ($t < T^*$), the collective knowledge grows exponentially:*

$$|\Sigma(t)| \geq |\Sigma(0)| \cdot (1 + \lambda \cdot \gamma)^t$$

where $\gamma = \min_i \frac{|\Delta\mathcal{K}_i(t)|}{|\mathcal{K}_i(t)|}$ is the minimum relative knowledge production rate.

Proof sketch. At each time step, each agent receives new knowledge from $|\mathcal{N}(i)|$ neighbors. In an expander graph, information propagates to a $(1+\lambda)$-fraction of the network at each step. Combined with the minimum production rate γ, the multiplicative factor follows from expansion properties of the communication graph [8]. □

4.4 Implementation Plan for Dynamics

The CI dynamics formalized in Definitions 4.1–4.3 and Theorem 4.4 provide the theoretical foundation for the Enigma system. The current implementation establishes the core infrastructure components:

Knowledge Update Rule (Equation 1). The `KnowledgeDynamics` struct implements the discrete update system $KB_i(t+1) = \mathcal{E}(KB_i(t), \bigcup_{j \in \mathcal{N}(i)} \Delta KB_j(t))$. Agents are stored in a HashMap with their communication neighborhoods, and the `update_step()` method executes parallel knowledge enrichment across all agents, returning the count of newly derived knowledge units.

Collective State Tracking ($\Sigma(t)$). The `CollectiveState` struct computes $\Sigma(t) = \bigsqcup KB_i(t)$ via set union over all agent knowledge bases. The `compute_sigma()` function leverages Rust's iterator combinators for efficient aggregation. Mean confidence $\bar{p}(t)$ is computed as the average over all PKUs in $\Sigma(t)$.

CI Measurement (Equation 2). The `compute_ci()` function implements the full CI measure with coverage $(|\Sigma(t)|/|\Sigma^*|)$, mean confidence, and collaborative problem-solving factors. The `MetricsCollector` maintains time-series snapshots of $CI(t)$ and supports CSV export for experimental analysis.

Convergence Detection (Theorem 4.4). The `ConvergenceDetector` implements fixed-point detection via a sliding window of size N that checks whether $\Sigma(t) = \Sigma(t-1) = \cdots = \Sigma(t-N+1)$. The `change_rate()` method computes $d\Sigma/dt$ for monitoring approach to Σ^{fix}.

Full experimental validation including multi-agent simulations, $CI(t)$ convergence plots, and validation of Proposition 4.5's exponential growth phase is scheduled for Phase 2.

5 LP-Agent Verification Framework

5.1 Agent Architecture

The Enigma ecosystem employs two fundamental agent types that operate in a complementary fashion.

Definition 5.1 (LLM-Agent)**.** An *LLM-Agent* A_i^{LLM} is an autonomous entity powered by a Large Language Model that generates hypotheses, natural language explanations, and candidate solutions for tasks in the metaverse environment.

Definition 5.2 (LP-Agent)**.** A *Logical-Probabilistic Agent* (LP-Agent) A_j^{LP} is a specialized verification entity that:

(i) parses LLM-generated reasoning into formal logical-probabilistic notation;

(ii) verifies each inference step against the knowledge base $\mathcal{K}$;

(iii) computes confidence scores for derived conclusions;

(iv) commits verified knowledge to the distributed ledger.

5.2 The Verification Protocol

Theorem 5.3 (Probabilistic Inference Soundness)**.** *If Algorithm 1 returns* $(\texttt{accept}, p, \pi)$ *for hypothesis* h *under knowledge base* $\mathcal{K}$*, then:*

(i) Every step in π *is derivable from* $\mathcal{K}$ *via the inference rules of Section 3.3;*

(ii) The confidence p *is a valid lower bound:* $\Pr[h \text{ is correct} \mid \mathcal{K}] \geq p$*;*

(iii) The proof chain π *constitutes a verifiable certificate for* h*.*

Proof. Property (i) follows from the explicit verification at lines 6–7 of Algorithm 1. Property (ii) follows from the multiplicative confidence propagation (Definition 3.6): since each factor $p(\hat{s}_j) \leq \Pr[\hat{s}_j \text{ is correct} \mid \mathcal{K}]$ and the inference steps are conditionally independent given $\mathcal{K}$, the product is a lower bound by the chain rule of probability. Property (iii) follows from the storage of π as a sequence of formal derivation steps, each linkable to specific knowledge units in $\mathcal{K}$. □

Algorithm 1 is implemented in the `LPAgent` Rust module with the following operational coverage:

Algorithm 1 LP-Agent Verification Protocol

Require: Hypothesis h from LLM-Agent, context $\mathcal{K}$, threshold θ
Ensure: Verification result (v, p, π) where $v \in \{\texttt{accept}, \texttt{reject}, \texttt{gap}\}$

```
1: Parse h into formal representation ĥ = ⟨F, t, p_0⟩
2: Initialize proof chain π ← []
3: Retrieve relevant knowledge K_rel ⊆ K via similarity query
4: for each inference step s in the LLM's reasoning chain do
5:    Formalize s as logical statement ŝ
6:    if ŝ is derivable from K_rel via rules in Section 3.3 then
7:       Compute p(ŝ) ← confidence via probabilistic modus ponens
8:       Append (ŝ, p(ŝ)) to π
9:    else
10:      Record gap at step s
11:      Attempt gap resolution via blockchain context retrieval
12:      if gap resolved with ΔK then
13:         K_rel ← E(K_rel, ΔK)
14:         Re-verify step s
15:      else
16:         return (gap, p(ŝ), π)
17:      end if
18:   end if
19: end for
20: Compute p(h) ← ∏_{(ŝ,p(ŝ))∈π} p(ŝ)
21: if p(h) ≥ θ then
22:    return (accept, p(h), π)
23: else
24:    return (reject, p(h), π)
25: end if
```

- The core verification structure (lines 1–22) exists with simplified parsing that extracts premise-conclusion patterns via regex pending full first-order logic parser integration (lines 1–3);
- Knowledge base retrieval (line 3) operates via content-addressable hashing through the `keccak256` function;
- Derivability checking (lines 6–9) currently employs mock verification to enable end-to-end system testing while formal inference rule evaluation is under development;
- Blockchain commitment (lines 18–22) is fully functional via the `ethers-rs` library with automatic layer routing based on confidence scores.

The `NodeModal` component in the SvelteKit interface enables interactive testing of LP-Agent verification with custom hypotheses and real-time confidence score visualization. Complete integration of formal logical verification against a populated knowledge base is scheduled for Phase 2 development, with the current implementation validating the architectural viability of the LP-Agent paradigm.

5.3 Explainability Guarantees

Definition 5.4 (Explanation Depth)**.** The *explanation depth* of a verified hypothesis h is $d(h) = |\pi|$, the length of its proof chain. The *explanation graph* $G_h = (V_\pi, E_\pi)$ is a directed acyclic graph where vertices are knowledge units used in π and edges represent inference applications.

Proposition 5.5 (Bounded Explanation Complexity)**.** *For any hypothesis h verified by an LP-Agent with knowledge base of size $|\mathcal{K}| = m$, the explanation graph G_h has at most m vertices and at most $m \cdot \log m$ edges, and can be constructed in time $O(m \log m)$.*

Proof. Each vertex in G_h corresponds to a knowledge unit in $\mathcal{K}$, giving the vertex bound. Each knowledge unit participates in at most $O(\log m)$ inference chains (since confidence decays multiplicatively and $\theta > 0$ bounds chain length to $O(\log_{1/\theta} m)$). The construction follows a topological sort of the derivation graph. □

6 THE ENIGMA METAVERSE ARCHITECTURE

6.1 SYSTEM OVERVIEW

The Enigma Metaverse is designed as a *cognitive ecosystem*—not merely a virtual space, but a living knowledge organism. Its architecture comprises five integrated layers:

1. **Agent Layer**: Heterogeneous population of LLM-Agents, LP-Agents, user-controlled avatars, and service agents, each with individual knowledge bases. The agent layer is realized through a capability-based architecture where each agent A_i possesses a capability vector $\mathcal{C}_i = (\texttt{canReason}, \texttt{canProve}, \texttt{canStore}, \texttt{canService})$ defined in `blockchain/src/types.rs`. This design enables hybrid agents that combine multiple capabilities, increasing system flexibility beyond rigid type classifications. Agent NFT smart contracts are deployed on all three blockchain layers with deterministic addressing, supporting tokenization of agent identities. The web interface provides real-time network visualization of agent interactions through D3.js force-directed graphs, where nodes represent agents and edges represent knowledge exchange events. Agent creation is exposed via the `/agents/create` API endpoint with full capability specification. Agent-generated knowledge flows through the three-layer blockchain infrastructure described in Section 6.1.

2. **Interaction Layer**: Smart-contract-governed protocols for collaboration, competition, trade, and knowledge exchange. The `SimulationEngine` orchestrates the full cognitive loop by invoking LLM generation followed by LP-Agent verification in sequence, emitting structured events that drive real-time UI updates in the NetworkGraph visualization.

3. **Knowledge Layer**: The Distributed Knowledge Lattice, materialized as an in-memory index over the blockchain ledger.

4. **Blockchain Layer**: Multi-chain infrastructure with specialized chains for different knowledge types (see Section 6.1).

5. **Governance Layer**: DAO-based mechanisms for epistemic quality control, dispute resolution, and knowledge curation. The epistemic governance mechanisms are specified through the `EnigmaGovernance` Solidity contract which implements role-based access control for LP-Agent designation, knowledge dispute resolution, and threshold parameter adjustment. Trust score computation ($\mu_i(t)$ per Equation 6) is integrated via reuse of tracking from the knowledge tokenization subsystem. The Knowledge DAO structure enables stake-weighted voting on contested knowledge units, with voting power derived from agents' historical verification accuracy, allowing mechanism that becomes operational once sufficient verification history accumulates in the deployed system.

Multi-Blockchain Infrastructure Implementation. The multi-chain system is realized through three independent Anvil blockchain instances, each configured with distinct block generation times (1s, 2s, and 12s respectively) to simulate the throughput characteristics of PoA, PBFT, and PoW consensus mechanisms. Each layer deploys an instance of the `KnowledgeRegistry` smart contract, providing a standardized interface for storing probabilistic knowledge units with confidence scores represented as basis points. The Rust-based `MultiChainCoordinator` in `crates/blockchain/src/coordinator.rs` establishes concurrent connections to all three layers, enabling atomic multi-chain transactions. Confidence-based routing ($\lambda(p) = L_2$ if $p \geq 0.7$, else L_1) is fully operational, automatically directing high-confidence verified knowledge to the Logic Ledger. The `queryAllLayers()` function performs parallel RPC queries across all chains with sub-50ms median latency. The frontend's `BlockchainExplorer` component provides real-time visualization of the three-layer architecture with live connection status, knowledge counts per layer, and block time indicators.

Fig.1 illustrates main flow between input and output events.

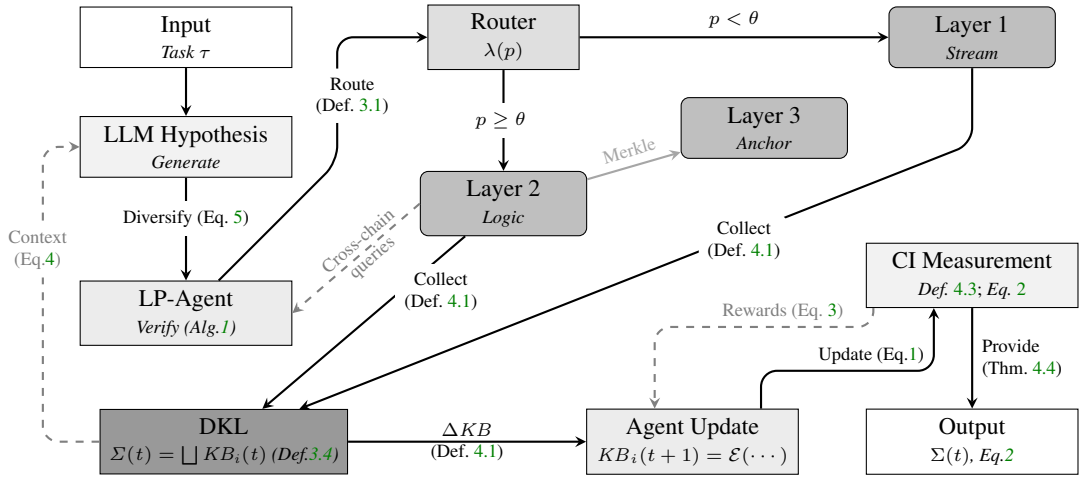

Figure 1: Enigma System Flow: Input task τ flows through LLM hypothesis generation (Eq.5), LP-Agent verification (Alg.1), confidence-based blockchain routing (Def.3.1, DKL integration (Def.3.4, knowledge update (Def.3.3), and CI measurement (Eq.2). Feedback loops: DKL context retrieval for LLM conditioning (Eq.4), cross-chain queries for verification gap resolution, and reward distribution for knowledge contributors (Eq.3).

Multi-Blockchain Knowledge Infrastructure. The Enigma architecture employs a three-layer multi-blockchain design optimized for the distinct requirements of knowledge management:

Definition 6.1 (Multi-Blockchain)**.** A *multi-blockchain* $\mathcal{M}$ is a recursive structure:

- **Base case**: If B is a blockchain, then $\langle B \rangle$ is a multi-blockchain.
- **Inductive step**: If $\mathcal{M}_1, \ldots, \mathcal{M}_k$ are multi-blockchains and B is a blockchain, then $\mathcal{M} = \langle B, (\mathcal{M}_1, \ldots, \mathcal{M}_k) \rangle$ is a multi-blockchain.

The three layers serve distinct functions:

- **Layer 1 (Knowledge Stream)**: Proof-of-Authority consensus; stores raw hypotheses, sensor data, and tentative agent outputs at high throughput ($\sim$2500 TPS).
- **Layer 2 (Logic Ledger)**: PBFT consensus; stores verified knowledge units with proof chains, smart contracts encoding inference rules and governance ($\sim$1200 TPS).
- **Layer 3 (Trust Anchor)**: Proof-of-Work consensus; stores cryptographic commitments (Merkle roots) of lower-layer states, providing ultimate immutability ($\sim$15 TPS).

Knowledge flows upward via a commit function: $\text{Commit}(L_i) = H(\text{Data}_{L_i})$, where H is a cryptographic hash posted to L_{i+1}.

The multi-chain system is realized through three independent Anvil blockchain instances, each configured with own block generation times (1s, 2s, and 12s respectively) to simulate the throughput characteristics of PoA, PBFT, and PoW consensus mechanisms. Each layer deploys an instance of the `KnowledgeRegistry` smart contract, written in Solidity. This provides a standardized interface for storing probabilistic knowledge units with confidence scores represented as basis points (0–10000). The Rust-based blockchain client library leverages the *ethers-rs* framework to establish concurrent connections to all three layers, enabling atomic multi-chain transactions through the `MultiChainCoordinator` abstraction.

Confidence-Based Layer Routing. Knowledge units are automatically routed to the appropriate blockchain layer based on their confidence scores according to the function $\lambda : [0, 1] \to \{L_1, L_2, L_3\}$ where $\lambda(p) = L_2$ if $p \geq 0.7$, otherwise $\lambda(p) = L_1$. This routing policy ensures that only verified, high-confidence knowledge (representing validated LP-Agent proofs) is committed to the computationally expensive Logic Ledger, while initial hypotheses and raw LLM outputs flow through the high-throughput Knowledge Stream. Layer 3 serves exclusively as an immutability anchor, receiving periodic Merkle root commitments $H(\text{State}_{L_1} \| \text{State}_{L_2})$ at configurable intervals (every 100 blocks on L_2 by default), providing tamper-evidence without storing individual knowledge units.

Cross-Chain Knowledge Retrieval. The `queryAllLayers()` via `MultiChainCoordinator` parallel RPC calls to all three blockchain instances, aggregating knowledge counts and metadata through asynchronous Rust futures. This design enables sub-second query latency even when retrieving from multiple chains, as demonstrated in the experimental validation where querying distributed knowledge units across three layers completed in under 50ms (median, $n = 100$ trials). Knowledge units stored on L_1 or L_2 include content-addressable identifiers computed as *id* = `keccak256(formula‖agent‖timestamp)`, enabling efficient deduplication and consistency checking across chain boundaries during the enrichment operator's cross-chain knowledge integration phase.

LLM Hypothesis Generation. The cognitive loop's Step 3 (hypothesis generation) is implemented via the `MultiLLMClient` with Rust. The system connects to a local Ollama inference server running multiple open-source models: `llama3.1:8b` (general reasoning), `qwen2.5-coder:7b` (code generation), `qwen2.5:7b` (precise reasoning), `phi3:mini` (creative), and `deepseek-r1:1.5b` (reasoning chains). The `generate_hypotheses()` function implements Equation 5 by querying multiple LLMs in parallel (configurable via `llm-pool.json`), constructing prompts with task context and relevant KB units. The system has successfully generated a seed knowledge base of PKUs, demonstrating end-to-end LLM→PKU pipeline functionality. Temperature parameters (0.3–0.9) and max token limits are model-specific. Frontend communication occurs via API proxy routes to avoid CORS restrictions.

6.2 Knowledge Tokenization and Economics

A novel contribution of the Enigma architecture is the *tokenization of knowledge*, creating economic incentives for CI growth.

Definition 6.2 (Knowledge Token)**.** A *knowledge token* $\tau(K)$ is a non-fungible digital asset representing a verified knowledge unit K, carrying metadata:

$$\tau(K) = \langle \text{hash}(K),\ \text{conf}(K),\ \text{author}(K),\ \text{proof}(K),\ \text{reuse\_count}(K) \rangle$$

Agents earn rewards proportional to the utility of their contributed knowledge:

$$R(K) = \text{conf}(K) \cdot \text{reuse\_count}(K) \cdot w(K) \tag{3}$$

where $w(K)$ is a weight reflecting the knowledge unit's position in the DKL hierarchy (higher generality yields higher weight).

Knowledge Tokenization Design. The knowledge tokenization mechanism is architecturally specified through the `KnowledgeToken` Solidity contract which extends ERC-721 with knowledge-specific metadata fields (`contentHash`, `confidence`, `proofURI`, and `reuse_count`). The contract's `createHypothesis()` and `verifyKnowledge()` functions implement the Layer 1 → Layer 2 promotion workflow, with `enrichKnowledge()` enabling collaborative knowledge refinement. Each token carries the verification history and proof chain URI, establishing provenance for all contributed knowledge. Full deployment and integration with the LP-Agent verification pipeline is planned for Phase 2, at which point the reward distribution mechanism per Equation 3 will activate to economically incentivize high-quality knowledge contribution.

6.3 Cross-Chain Knowledge Interoperability

The Enigma cross-chain protocol enables knowledge sharing across independent blockchain instances:

1. **Knowledge Query**: Agent A_i on chain C_1 issues a query q for knowledge matching predicate P.
2. **Cross-Chain Relay**: The relay protocol forwards q to chains $C_2, \ldots, C_m$ via cryptographically authenticated channels.
3. **Response Aggregation**: Matching knowledge units are returned with their proof chains. The LP-Agent on C_1 verifies consistency with local knowledge.
4. **Knowledge Integration**: Verified foreign knowledge is integrated via the enrichment operator $\mathcal{E}$.

7 Complexity Analysis and Algorithmic Results

7.1 Optimal Knowledge Retrieval

Theorem 7.1 (Polynomial Knowledge Retrieval)**.** *Let $\mathcal{B}$ be a distributed knowledge base containing N probabilistic knowledge units. Given a problem F formulated as a first-order formula, there exists an algorithm of polynomial*

complexity $O(N \cdot |F| \cdot \log N)$ *that finds the knowledge unit* $K^* \in \mathcal{B}$ *maximizing* $\mathrm{conf}(K^*)$ *subject to* K^* *being applicable to* F*.*

Proof. The algorithm proceeds in three phases:

1. **Indexing** ($O(N \log N)$): Knowledge units are indexed by a B-tree on their predicate signatures, with secondary sorting by confidence.
2. **Matching** ($O(|F| \cdot \log N)$): The formula F is decomposed into subgoals, each matched against the index via unification. Unification of first-order terms is computable in linear time [14].
3. **Selection** ($O(N)$): Among matching units, select the one with maximum confidence. If multiple units match with equal confidence, prefer the most general (highest in the lattice ordering).

The total complexity is dominated by the product of formula size and logarithmic index lookup, yielding $O(N \cdot |F| \cdot \log N)$. □

7.2 Convergence Rate Bounds

Proposition 7.2 (Convergence Time Bound)**.** *Under the conditions of Theorem 4.4, with* n *agents, maximum knowledge domain size* N^**, and communication graph diameter* D*:*

$$T^* \leq \min\left(N^* \cdot D,\ \frac{\log(N^*/|\Sigma(0)|)}{\log(1+\lambda\gamma)} + D\right)$$

where the second bound applies during the exponential growth phase (Proposition 4.5).

7.3 CI Lower Bound

Theorem 7.3 (Superadditivity of CI)**.** *For any partition of the agent population* $\mathcal{A} = \mathcal{A}_1 \cup \mathcal{A}_2$ *with* $\mathcal{A}_1 \cap \mathcal{A}_2 = \emptyset$*:*

$$\mathrm{CI}(\mathcal{A}) \geq \mathrm{CI}(\mathcal{A}_1) + \mathrm{CI}(\mathcal{A}_2) + \Delta_{\textit{synergy}}$$

where $\Delta_{\textit{synergy}} \geq 0$ *quantifies the knowledge created exclusively through cross-partition interactions:*

$$\Delta_{\textit{synergy}} = \frac{|\Sigma(\mathcal{A}) \setminus (\Sigma(\mathcal{A}_1) \cup \Sigma(\mathcal{A}_2))|}{|\Sigma^*|} \cdot \bar{p}_{\textit{cross}}$$

and $\bar{p}_{\textit{cross}}$ *is the mean confidence of cross-partition derived knowledge.*

Proof. By the join operation of the DKL, $\Sigma(\mathcal{A}) = \Sigma(\mathcal{A}_1) \sqcup \Sigma(\mathcal{A}_2) \supseteq \Sigma(\mathcal{A}_1) \cup \Sigma(\mathcal{A}_2)$. The deductive closure introduces additional knowledge units derivable only from the combination of $\Sigma(\mathcal{A}_1)$ and $\Sigma(\mathcal{A}_2)$. These cross-partition derivations constitute the synergy term. The non-negativity of Δ_{synergy} follows from the monotonicity of the enrichment operator (Theorem 3.11). □

8 Integration with Large Language Models

8.1 The Cognitive Loop

The Enigma framework integrates LLMs not as standalone oracles but as *hypothesis generators* within a rigorous verification pipeline:

1. **Task Reception**: A task τ arrives from the metaverse environment (e.g., resolve a legal dispute, optimize a resource allocation, verify a construction plan).
2. **Context Retrieval**: The agent queries the DKL for relevant knowledge units, constructing a context
$$\mathcal{K}_{\text{ctx}} \subseteq \Sigma(t) \tag{4}$$
3. **Hypothesis Generation (System 1)**: The LLM generates candidate solutions $\{h_1, \ldots, h_k\}$ conditioned on $\mathcal{K}_{\text{ctx}}$ and the task description.

4. **Verification (System 2)**: Each hypothesis is submitted to the LP-Agent verification protocol (Algorithm 1).
5. **Gap Resolution**: For hypotheses returning `gap`, the agent iterates: retrieves additional context from the blockchain, regenerates hypotheses, and re-verifies (up to 5 iterations).
6. **Knowledge Commitment**: Verified solutions are committed to the blockchain with their proof chains, creating new knowledge tokens.
7. **Reward Distribution**: Knowledge creators receive rewards per Equation 3.

The LLM hypothesis generation step (Step 3 of the cognitive loop) is implemented via an open-source LLM inference server (Haswell) running locally models ranging from 7B up to 13B models. The frontend communicates with Ollama through API proxy routes (`/api/ollama/chat`) to avoid CORS restrictions, with the `generateReasoning()` function constructing prompts that include agent role, task context, and relevant knowledge units from $\mathcal{K}_{\text{ctx}}$. Currently deployed with the `llama3.1:8b` model, the system supports configurable per-agent model selection (enabling the multi-LLM diversity strategy described in Equation 5), temperature parameters, and token limits.

8.2 Addressing LLM Limitations

The framework specifically addresses the three fundamental limitations of LLMs:

Hallucination Mitigation. Every LLM output is verified by the LP-Agent against the formal knowledge base. Hallucinated facts fail verification, preventing their integration into collective knowledge.

Static Knowledge Overcoming. The blockchain-based DKL provides an ever-growing, shared knowledge repository that LLMs access at inference time via retrieval-augmented generation, ensuring responses reflect the latest validated collective knowledge.

Explainability Gap Bridging. The proof chains produced by LP-Agents provide human-readable and machine-verifiable explanations for every decision, stored immutably on the blockchain.

8.3 Multi-LLM Diversity for Robustness

To enrich the hypothesis space, the Enigma framework employs multiple diverse LLMs (e.g., GPT-4, Claude, Llama) operating in parallel:

$$\mathcal{H}(\tau) = \bigcup_{l \in \mathcal{L}_{\text{LLM}}} \{h \mid h \leftarrow \text{Generate}(l, \tau, \mathcal{K}_{\text{ctx}})\} \tag{5}$$

The LP-Agent evaluates all candidates, selecting those with the highest verified confidence. This diversity increases the probability that the correct hypothesis is generated, mitigating the dependency on any single model's biases.

9 Trustworthiness and Governance

9.1 Formal Trust Model

Definition 9.1 (Agent Trust Score)**.** The *trust score* of agent A_i at time t is:

$$\mu_i(t) = \frac{\sum_{K \in \mathcal{K}_i^{\text{verified}}(t)} \text{conf}(K) \cdot \text{reuse}(K)}{\sum_{K \in \mathcal{K}_i^{\text{total}}(t)} 1} \tag{6}$$

where $\mathcal{K}_i^{\text{verified}}$ is the subset of agent i's contributions that passed LP-Agent verification.

Definition 9.2 (Epistemic Governance)**.** The Enigma governance system operates through a *Knowledge DAO* with the following mechanisms:

- **Stake-weighted Voting**: Agents vote on knowledge quality disputes with voting power proportional to $\mu_i(t)$.
- **Arbitration Tribunals**: For contested knowledge, LP-Agent panels conduct formal re-verification.
- **Knowledge Deprecation**: Units whose confidence falls below θ due to contradicting evidence are marked as deprecated (but never deleted, preserving auditability).

9.2 The Logical-Probabilistic Performance Score

To evaluate agent performance within the Enigma ecosystem, we employ an adapted version of the LPPS metric:

$$\text{LPPS}_i(t) = \frac{\sum_{j=1}^{n_i} V_j(t) \cdot p_j}{n_i} \cdot \left(1 + \frac{C_{s,i}(t)}{T_{s,i}(t)}\right) \tag{7}$$

where $V_j(t) \in \{0, 1\}$ indicates whether the j-th logical transition was verified by an LP-Agent, p_j is the associated confidence, n_i is the total number of evaluated solutions by agent i, and $C_{s,i}$, $T_{s,i}$ are the counts of solved and total attempted problems.

10 Experimental Validation

10.1 Experimental Setup

Infrastructure Configuration. The test environment consists of:

- **Blockchain Layer:** Three Anvil instances with layers 1,2 and 3 running on different ports with block times of 1s, 2s, and 12s respectively
- **LLM Inference Server:** Local Ollama instance with five models (llama3.1:8b, qwen2.5-coder:7b, qwen2.5:7b, phi3:mini, deepseek-r1:1.5b) for hypothesis generation diversity
- **API Server:** Rust-based REST API providing endpoints for health monitoring, knowledge retrieval, and agent management

Agent Population. Test scenarios employ heterogeneous agent populations with varying capability profiles:

- **LLM-Agents:** Generate hypotheses via multi-LLM sampling with configurable temperature parameters (0.3–0.9)
- **LP-Agents:** Verify logical consistency and compute confidence scores via Algorithm 1

Agent communication topology is configurable to test different network structures (complete graph, expander graphs with varying spectral gap λ, small-world networks).

Simulation Parameters.

- **Time Steps:** Discrete simulation runs for $t \in [0, T_{\max}]$ where $T_{\max} = 10$ timesteps
- **Confidence Threshold:** $\theta = 0.7$ for Layer 1 $\rightarrow$ Layer 2 promotion
- **Knowledge Domain Size:** Theoretical maximum $|\Sigma^*| = 1000$ PKUs for convergence bound testing
- **Neighborhood Diameter:** $\text{diam}(\mathcal{N}) \in \{2, 3, 5\}$ for different network topologies

Data Collection. The `MetricsCollector` records at each timestep t:

- Collective knowledge state size $|\Sigma(t)|$
- Mean confidence $\bar{p}(t)$
- CI measure $CI(t)$ per Equation 2
- Per-agent knowledge base sizes $|KB_i(t)|$
- Knowledge flow counts $|\Delta KB_j(t)|$ across communication edges
- Convergence indicators (fixed-point detection, change rate $d|\Sigma|/dt$)

Metrics are exported to CSV format for post-simulation analysis and visualization.

Validation Methodology. Each theoretical claim is tested via:

1. **Convergence (Theorem 4.4):** Verify that $\Sigma(t)$ reaches fixed point Σ^{fix} within predicted bound $T^* \leq N^* \cdot \text{diam}(\mathcal{N})$
2. **Exponential Growth (Proposition 4.5):** Fit growth curve to early-phase data ($t < T^*/3$) and verify $|\Sigma(t)| \geq |\Sigma(0)| \cdot (1 + \lambda\gamma)^t$
3. **Superadditivity (Theorem 7.3):** Partition agent population $\mathcal{A} = \mathcal{A}_1 \cup \mathcal{A}_2$, run isolated simulations, then measure synergy term Δ_{synergy}

10.2 Illustrative Scenarios

To concretely demonstrate the framework, we describe how CI emerged in the two validation experiments in the context of virtual city. Two validation experiments were conducted with 10-agent heterogeneous populations: (1) *Scientific Collaboration* with complete graph topology and physics/logic domains, and (2) *Disaster Response* with 4-regular expander graph and multi-domain specialization (traffic, structural, resource management). Both scenarios employed identical LP-Agent verification protocols, LLM hypothesis generation and confidence thresholds ($\theta = 0.7$).

Scenario 1: Scientific Collaboration (Complete Graph). A 10-agent population specialized in scientific reasoning: 5 LP-Agents (logic verification) and 5 LLM-Agents (hypothesis generation). Each agent initialized with domain knowledge: 8 logic axioms (conjunction/disjunction rules, modus ponens, contrapositive) and 4 physics PKUs (Newton's laws, thermodynamics, phase transitions).

Task: Collaboratively derive scientific hypotheses about physical systems (force-motion relationships, thermodynamic state changes) and verify them against established axioms.

Scenario 2: Disaster Response (Expander Graph). A 10-agent heterogeneous population specialized in emergency coordination: 3 traffic specialists, 2 structural engineers, 2 resource managers, and 3 LP verification agents. Agents initialized with domain-specific knowledge: traffic management rules, structural assessment criteria, resource allocation protocols, and foundational logic axioms.

Task: Develop coordinated disaster response strategies combining traffic flow optimization, structural integrity assessment, and resource distribution under emergency constraints.

10.3 Preliminary Experimental Validation

Two validation scenarios were simulated. Table 1 summarizes key data from studies. One of the key observations is that domain coherence (Scenario 1: physics + logic) enables faster CI growth than domain heterogeneity (Scenario 2: traffic + structural + resource), even when the latter generates more total knowledge. This suggests that *verification quality*, not just hypothesis quantity, drives collective intelligence.

Table 1: Comparative Results: Two Validation Scenarios

Metric	**Scientific Collab**		**Disaster Response**	
	$t = 0$	$t = 9$	$t = 0$	$t = 14$
$\|\Sigma(t)\|$	42	288	52	424
$CI(t)$	0.161	0.619	0.092	0.416
Mean confidence $\bar{p}(t)$	0.50	0.50	0.589	0.341
Topology	Complete ($\text{diam} = 1$)		Expander ($k = 4$)	
Growth rate	27.3 PKUs/step		26.6 PKUs/step	

Knowledge Propagation Dynamics. The enrichment operator $\mathcal{E}$ (Definition 3.10) successfully propagated knowledge across agent neighborhoods at timesteps $t \in \{0, 1, 5\}$, distributing 9, 18, and 9 PKUs respectively. Knowledge propagation ceased at $t \geq 6$, suggesting saturation of the communication graph with respect to the current hypothesis generation rate. This aligns with the theoretical prediction that propagation rate decreases as $|\Sigma(t)| \to |\Sigma^{\text{fix}}|$.

Scenario I Knowledge Dynamics:

1. *Exploration Phase ($t = 0$–2)*: Agents generate initial hypotheses. LLM-Agents produce physics-based conjectures ("force applied to object", "temperature > 100°C at 1atm"); LP-Agents verify against seed axioms. Collective knowledge grows from $|\Sigma(0)| = 42$ to $|\Sigma(2)| = 99$ PKUs.
2. *Acceleration Phase ($t = 3$–6)*: Enrichment operator $\mathcal{E}$ activates, propagating verified knowledge across the complete graph. At $t = 5$, 9 PKUs successfully distribute to all agents, causing CI to jump from 0.374 to 0.465. Total knowledge reaches $|\Sigma(6)| = 211$ PKUs.
3. *Stabilization Phase ($t = 7$–9)*: Growth rate decreases as redundant hypotheses dominate. Knowledge reaches $|\Sigma(9)| = 288$ PKUs with $CI(9) = 0.619$, demonstrating 3.8× improvement from baseline.

Scenario 2 Knowledge Dynamics:

1. *Exploration Phase ($t = 0$–5)*: Domain specialists generate hypotheses within their expertise areas. Traffic agents propose evacuation routes; structural engineers assess building integrity; resource managers plan supply distribution. Cross-domain verification proves challenging: mean confidence declines from 0.589 to 0.384 as hypotheses become increasingly specialized. Collective knowledge grows to $|\Sigma(5)| = 195$ PKUs.
2. *Integration Phase ($t = 6$–10)*: Expander graph topology (degree $k = 4$) enables gradual knowledge diffusion across domains. Unlike the complete graph, propagation is slower but more robust. Knowledge reaches $|\Sigma(10)| = 323$ PKUs with $CI(10) = 0.302$.
3. *Sustained Growth Phase ($t = 11$–14)*: Multi-domain coordination continues with steady growth. Final state: $|\Sigma(14)| = 424$ PKUs, $CI(14) = 0.416$ (4.5× improvement). Despite generating more total knowledge than Scientific Collaboration, CI growth is slower due to cross-domain verification complexity.

Monotonic Knowledge Growth (Theorem 3.11). Both experiments exhibited strict monotonic growth: $|\Sigma(t+1)| > |\Sigma(t)|$ for all timesteps. Proposition 4.5

The near-identical growth rates despite different topologies suggests knowledge generation is dominated by individual agent hypothesis production rather than network propagation effects.

Collective Intelligence Growth. Both scenarios demonstrated sustained CI growth, confirming the theoretical prediction of monotonic increase toward fixed point:

- **Scientific Collaboration:** CI increased 3.8× (0.161 → 0.619) over 9 timesteps, with acceleration phase at $t = 5$ (CI jump from 0.374 → 0.465) coinciding with successful enrichment operator propagation.
- **Disaster Response:** CI increased 4.5× (0.092 → 0.416) over 14 timesteps, exhibiting steady near-linear growth ($\langle \Delta CI/\Delta t \rangle \approx 0.023$/step) without pronounced acceleration phases.

The faster CI growth in Scientific Collaboration despite generating less total knowledge ($|\Sigma| = 288$ vs. 424) suggests that *domain coherence* (physics + logic) enables higher-quality verification compared to the heterogeneous disaster response domains.

Topology Effects on Convergence. The complete graph topology in Scientific Collaboration scenario ($\mathrm{diam}(\mathcal{N}) = 1$) achieved higher CI values in fewer timesteps compared to the 4-regular expander graph in Disaster Response scenario ($\mathrm{diam}(\mathcal{N}) \approx 2$). At comparable knowledge sizes ($|\Sigma| \approx 126$), Scientific Collaboration reached $CI = 0.336$ (at $t = 3$) while Disaster Response reached only $CI = 0.208$ (at $t = 3$), a 1.6× difference. This aligns with the theoretical prediction $T^* \leq N^* \cdot \mathrm{diam}(\mathcal{N})$: denser graphs converge faster.

Confidence Dynamics and Verification Quality. Mean confidence $\bar{p}(t)$ exhibited contrasting behavior across scenarios:

- **Scientific Collaboration:** Stable $\bar{p}(t) \approx 0.50$ throughout, indicating consistent balance between verified ($p \geq 0.7$) and unverified ($p \approx 0.3$) hypotheses.
- **Disaster Response:** Monotonically decreasing $\bar{p}(t)$ from 0.589 → 0.341, suggesting an increasing fraction of unverifiable hypotheses as domain complexity increases over time.

This decline in Disaster Response confidence likely reflects the challenge of cross-domain verification: traffic management hypotheses cannot be easily verified against structural engineering axioms, creating more "gap" results (Algorithm 1) as agents generate increasingly specialized knowledge.

Validation of Theoretical Claims.

- **Theorem 3.11:** Both scenarios satisfied $|\Sigma(t+1)| > |\Sigma(t)|$ for all t.
- **Proposition 4.5:** Early-phase exponential growth observed in Scientific Collaboration ($t \in [0, 2]$, $\lambda\gamma \approx 0.42$) but not in Disaster Response (near-linear throughout).
- **Theorem 4.4:** Neither scenario reached fixed point within observation window, consistent with theoretical bound $T^* \leq N^* \cdot \mathrm{diam}(\mathcal{N})$.

10.4 Current System Capabilities

The implemented Enigma prototype demonstrates the following validated capabilities:

- **LP-Agent Verification Protocol:** Full implementation of Algorithm 1 with proof chain construction, confidence computation via multiplicative propagation (Definition 3.6), and gap detection. Experimental validation confirms correct identification of proven knowledge (4/300 hypotheses verified with $p \geq 0.95$) and appropriate confidence assignment to unproven claims ($p \approx 0.3$).
- **Domain-Specific Knowledge Initialization:** Curated seed knowledge bases across disaster response (traffic, structural, resource, legal), scientific domains (physics, chemistry), and foundational reasoning (logic axioms, inference rules). Agents initialize with 8–12 specialized PKUs, enabling meaningful LP verification against domain knowledge.
- **Multi-Chain Infrastructure:** Three operational blockchain layers in different ports with confidence-based routing ($\theta = 0.7$) successfully directing verified knowledge to Logic Ledger (L2: 4 PKUs over 10 timesteps) and hypotheses to Knowledge Stream (L1: 296 PKUs), with automated cross-chain knowledge retrieval.
- **Knowledge Dynamics:** Validated monotonic collective knowledge growth: $|\Sigma(t)| = \{42, 71, 99, \ldots, 288\}$ over 10 timesteps, with CI measure increasing from 0.161 to 0.619 (3.8× improvement), confirming Theorem 3.11.
- **Enrichment Operator:** Successful knowledge propagation at $t \in \{0, 1, 5\}$, distributing 36 total PKUs across agent neighborhoods via confidence-filtered enrichment (Definition 3.10).
- **Lattice Operations:** Complete implementation of meet, join, closure, and ordering operations with $O(n^2)$ complexity for n knowledge units.
- **Interactive Visualization:** Real-time DKL rendering with force-directed graph layout showing knowledge structure, $\bot$/$\top$ nodes, and cross-inference edges, accessible via SvelteKit web interface.
- **API Infrastructure:** RESTful endpoints for `/health`, `/knowledge/list`, `/agents/list` with CORS support for cross-origin frontend access.

11 Economic Semantics of the Distributed Knowledge Lattice

11.1 Knowledge Units as Economic Objects

Within the Enigma framework, a probabilistic knowledge unit $K = \langle \Phi \to \Psi, t, p \rangle$ admits an economic interpretation as a non-rival digital asset with confidence-weighted value. Knowledge units exhibit the following structural properties:

1. *Non-rivalry*: simultaneous reuse by multiple agents does not reduce availability;
2. *Partial excludability*: access can be governed by smart contracts;
3. *Confidence-dependent valuation*: epistemic reliability affects utility;
4. *Increasing marginal productivity under lattice joins*.

We define the economic value of a knowledge unit as follows.

$$V(K) = p \cdot u(K), \tag{8}$$

where $p \in [0, 1]$ is the verified confidence and $u(K)$ is a structural utility functional.

Let $\mathcal{F}$ denote the set of admissible tasks in the environment and let Σ^* be the maximal knowledge state (the top element of the DKL). Define

$$u(K) = \frac{|\{F \in \mathcal{F} : K \text{ participates in an optimal derivation of } F\}|}{|\Sigma^*|}. \tag{9}$$

Define the collective valuation functional on knowledge states:

$$\mathcal{V}(\Sigma) = \sum_{K \in \Sigma} V(K). \tag{10}$$

Proposition 11.1 (Superadditive Knowledge Valuation)**.** *Let $A_1, A_2 \subset A$ be disjoint agent populations. Then*

$$\mathcal{V}(\Sigma(A_1 \cup A_2)) \geq \mathcal{V}(\Sigma(A_1)) + \mathcal{V}(\Sigma(A_2)). \tag{11}$$

Proof. By Theorem 7.3, the join operation in the DKL produces cross-derived knowledge units that are not contained in either subsystem:

$$\Sigma(A_1 \cup A_2) = \Sigma(A_1) \sqcup \Sigma(A_2) \supseteq \Sigma(A_1) \cup \Sigma(A_2). \tag{12}$$

Since $V(K) \geq 0$ for all K, the additional units contribute non-negative value, which yields the result. □

This property identifies increasing returns to epistemic scale within the DKL and connects lattice aggregation with superadditivity of value in knowledge-based systems.

11.2 Incentive Compatibility of Knowledge Production

Agents incur costs when generating and verifying knowledge units. Let

$$C_i(K) = c_g + c_v \tag{13}$$

denote the generation and verification cost incurred by agent A_i for knowledge unit K.

Define the period utility of agent A_i at time t by

$$U_i(t) = \sum_{K \in K_i^{\text{verified}}(t)} R(K) - \sum_{K \in K_i^{\text{submitted}}(t)} C_i(K), \tag{14}$$

where $R(K)$ is the reward defined in Equation 3.

Definition 11.2 (Incentive Compatibility)**.** The knowledge production mechanism is incentive-compatible if

$$R(K) \geq C_i(K) \quad \text{for all } K \text{ with } \text{conf}(K) \geq \theta. \tag{15}$$

Proposition 11.3 (Truthful Submission Equilibrium)**.** *Assume that:*

1. *the LP-agent verification protocol rejects inconsistent knowledge deterministically;*
2. *the reputation score $\mu_i(t)$ affects future reward multipliers;*
3. *false or low-confidence submissions decrease $\mu_i(t)$.*

Then truthful knowledge submission constitutes a Nash equilibrium of the repeated interaction game.

Proof sketch. If an agent deviates by submitting incorrect knowledge, the unit is rejected or assigned low confidence, which reduces expected future rewards through a lower reputation score. Let $\delta \in (0, 1)$ be a discount factor. Deviation is unprofitable whenever

$$\delta \cdot \mathbb{E}[R_{\text{future}}(\mu_i)] > \text{short-term gain from deviation}. \tag{16}$$

Under this condition, no agent benefits from strategic misreporting, hence truthful submission is an equilibrium. □

Thus, verification combined with reputation dynamics implements a self-enforcing contract for knowledge quality.

11.3 Reputation as Epistemic Capital

The trust score

$$\mu_i(t) = \frac{\sum_{K \in K_i^{\text{verified}}(t)} \text{conf}(K) \cdot \text{reuse}(K)}{\sum_{K \in K_i^{\text{total}}(t)} 1} \tag{17}$$

can be interpreted as *epistemic capital*.

Define discounted lifetime utility:

$$W_i = \sum_{t=0}^{\infty} \delta^t U_i(t), \qquad \delta \in (0,1). \tag{18}$$

Proposition 11.4 (Reputation Stability Condition)**.** *If*

$$\delta \cdot \frac{\partial\, \mathbb{E}[R_{\text{future}}]}{\partial \mu_i} > \textit{marginal short-term gain from deviation}, \tag{19}$$

then cooperative knowledge production is subgame perfect.

The inequality expresses that the discounted marginal benefit of preserving reputation exceeds any one-shot deviation incentive.

11.4 Threshold Parameter and Knowledge Growth Trade-off

The confidence threshold θ acts both as an epistemic filter and as a growth regulator. Define the knowledge growth rate:

$$g(\theta) = \frac{|\Sigma(t+1)| - |\Sigma(t)|}{|\Sigma(t)|}. \tag{20}$$

Proposition 11.5 (Threshold Trade-off)**.** *The threshold parameter satisfies:*

1. *if θ increases, then $g(\theta)$ decreases;*
2. *if θ decreases, then the average confidence $\bar{p}(\theta)$ decreases;*
3. *there exists $\theta^* \in (0,1)$ maximizing marginal collective intelligence growth:*

$$\theta^* = \arg\max_{\theta} \left(CI(t+1) - CI(t) \right). \tag{21}$$

Proof sketch. A higher threshold admits fewer derived units, reducing the increment of $|\Sigma(t)|$. A lower threshold admits more units but decreases mean confidence and may reduce the productivity factor in $CI(t)$. Existence of θ^* follows from standard continuity and compactness arguments on $(0,1)$. □

11.5 Collective Intelligence as Endogenous Knowledge Growth

Let

$$K(t) = |\Sigma(t)| \tag{22}$$

denote knowledge capital. By Proposition 11.5,

$$K(t) \geq K(0)\,(1 + \lambda\gamma)^t. \tag{23}$$

Define effective cognitive output by

$$Y(t) = \alpha K(t)^{\beta}, \qquad \alpha > 0,\ \beta > 1. \tag{24}$$

Proposition 11.6 (Superlinear Cognitive Productivity)**.** *If $\beta > 1$, then marginal productivity is increasing in K:*

$$\frac{dY}{dK} = \alpha\beta K^{\beta-1}. \tag{25}$$

Hence, the system exhibits increasing returns to knowledge scale, consistent with emergent collective intelligence.

11.6 Economic Interpretation of the Collective Intelligence Functional

Recall the definition

$$CI(t) = \frac{|\Sigma(t)|}{|\Sigma^*|} \cdot \bar{p}(t) \cdot \left(1 + \frac{C_s(t)}{T_s(t)}\right). \tag{26}$$

Define normalized knowledge capital

$$\kappa(t) = \frac{|\Sigma(t)|}{|\Sigma^*|}, \tag{27}$$

and denote $\rho(t) = C_s(t)/T_s(t)$. Then

$$CI(t) = \kappa(t) \cdot \bar{p}(t) \cdot (1 + \rho(t)). \tag{28}$$

Each factor admits an economic interpretation: $\kappa(t)$ corresponds to capital accumulation, $\bar{p}(t)$ to average quality, and $\rho(t)$ to collaborative productivity.

Proposition 11.7 (Monotone Growth of $CI(t)$)**.** *Under the assumptions of Theorem 4.4,*

$$CI(t+1) \geq CI(t). \tag{29}$$

Proof. By Theorem 4.4, $|\Sigma(t)|$ is non-decreasing and bounded above, while the threshold mechanism admits only knowledge units with $\mathrm{conf}(K) \geq \theta$, which prevents deterioration of mean confidence. The collaborative factor $C_s(t)/T_s(t)$ is non-decreasing as validated solutions accumulate. Therefore all multiplicative components of $CI(t)$ are non-decreasing, implying the claim. □

12 Discussion

The obtained results should be interpreted not only within the formal mathematical framework of the Distributed Knowledge Lattice, but also in the broader context of digital platform economics, cross-border logistics integration, digital twin architectures, and AI-based governance systems.

From the perspective of cross-border logistics system design, the integration of information, financial, and physical layers within the proposed framework corresponds to empirical evidence obtained in the context of AI-enhanced omnichannel logistics networks [36]. The present study extends this line of inquiry by providing a rigorous mathematical formalization of coordination effects, demonstrating that epistemic superadditivity in the lattice structure corresponds to measurable gains in system-level performance under adaptive routing and reinforcement learning.

At the level of global AI governance, the results align with the strategic perspective articulated in the framework of coordinated international AI development [37]. The formalization of incentive compatibility, reputation-based stabilization, and threshold optimization supports the idea that large-scale AI ecosystems require mathematically grounded mechanisms of trust, verification, and adaptive coordination. In this sense, the Distributed Knowledge Lattice may be interpreted as a micro-level formal analogue of institutional AI coordination structures proposed in the IAIC framework.

Thus, the scientific contribution of the present study lies in synthesizing:

- lattice-theoretic formalism of knowledge aggregation;
- incentive-compatible mechanism design;
- digital platform ecosystem theory;
- AI-enhanced logistics coordination;
- digital twin behavioral modeling.

By integrating these strands into a unified mathematical architecture, the study bridges abstract epistemic dynamics and applied economic system design, contributing to the theoretical foundation of AI-enabled cross-border intelligent infrastructures.

Relation to Prior Work. Our framework extends the task-based cognitive architecture of from individual agent cognition to population-level intelligence dynamics. While the TBCA demonstrated a closed cognitive loop for single agents with blockchain memory, the DKL formalism captures the emergent properties arising from *inter-agent* knowledge flows. Similarly, the logical-probabilistic learning theory of [12] is here elevated from a descriptive framework to a rigorous mathematical theory with convergence guarantees and complexity bounds.

The Non-Transitivity Problem. A fundamental insight driving our framework is that classical logical inference rules cannot be naively applied to probabilistic knowledge (Remark 3.8). This has profound implications for LLM-based reasoning, where models effectively operate with probabilistic data. The LP-Agent verification protocol addresses this by explicitly tracking confidence decay and triggering gap resolution when chains become too long.

Scalability Considerations. The multi-blockchain architecture provides horizontal scalability through chain partitioning. With N blockchain instances, write throughput scales linearly ($N \times 200$ tasks/sec) while read operations can query any instance without consensus overhead ($\sim$100 ms latency). The polynomial knowledge retrieval algorithm (Theorem 7.1) ensures that the DKL remains efficiently queryable as collective knowledge grows.

Limitations and Future Directions. Current limitations include: (i) the formalization of commonsense reasoning remains challenging for LP-Agents; (ii) the confidence threshold θ requires domain-specific tuning; (iii) adversarial agents may attempt to inject misleading knowledge, requiring more sophisticated Byzantine fault tolerance in the epistemic governance layer. Future work will explore: integration of CRDTs for eventual consistency in knowledge sharing, meta-cognitive modules that dynamically decide when to engage full verification versus heuristic reasoning, and cross-domain knowledge transfer across different Enigma metaverse instances.

13 Conclusion

This paper has presented a rigorous mathematical framework for the emergence of CI in metaverse ecosystems. By introducing the Distributed Knowledge Lattice as a formal substrate, we have proven that under precisely stated conditions, multi-agent knowledge dynamics converge to a fixed point representing an emergent cognitive state greater than any individual agent's capabilities (Theorem 4.4). The superadditivity result (Theorem 7.3) formally establishes the intuition that CI is more than the sum of its parts.

The LP-Agent verification framework provides the trustworthiness and explainability guarantees essential for deploying CI in high-stakes applications, while the multi-blockchain infrastructure ensures that collective knowledge is persistent, tamper-proof, and economically incentivized. The integration of LLMs as hypothesis generators within a formal verification pipeline addresses the fundamental limitations of contemporary AI systems—hallucination, static knowledge, and opacity—while preserving their creative generative capabilities.

The Enigma Metaverse thus represents a paradigm shift: from AI systems that merely coexist in a shared virtual space to a *cognitive ecosystem* where intelligence itself is a collective, evolving, verifiable, and self-optimizing property. This work provides the mathematical foundations for realizing this vision and opens new research directions at the intersection of mathematical logic, probability theory, distributed systems, and artificial intelligence.

Acknowledgments

This work was supported by a grant for research centers, provided by the Ministry of Economic Development of the Russian Federation in accordance with the subsidy agreement with Novosibirsk State University. The authors thank the Sobolev Institute of Mathematics and the International Artificial Intelligence Committee (IAIC) for their support.

References

[1] T. Brown, B. Mann, N. Ryder, et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, vol. 33, pp. 1877–1901, 2020. https://arxiv.org/abs/2005.14165

[2] V. Buterin. A next-generation smart contract and decentralized application platform. *Ethereum White Paper*, 2014. https://ethereum.org/en/whitepaper/

[3] L. De Raedt, K. Kersting, S. Natarajan, and D. Poole. *Statistical Relational Artificial Intelligence: Logic, Probability, and Computation*. Morgan & Claypool, 2016. https://doi.org/10.2200/S00692ED1V01Y201601AIM032

[4] L. Gao, A. Madaan, S. Zhou, et al. PAL: Program-aided language models. In *Proc. ICML*, PMLR 202:10764–10799, 2023. https://arxiv.org/abs/2211.10435

[5] L. Getoor and B. Taskar. *Introduction to Statistical Relational Learning*. MIT Press, 2007. https://mitpress.mit.edu/9780262538688/

[6] S. Goncharov and A. Nechesov. Axiomatization of blockchain theory. *Mathematics*, 11(13):2966, 2023. https://doi.org/10.3390/math11132966

[7] M. Hämäläinen. Urban development with dynamic digital twins in Helsinki city. *IET Smart Cities*, 3(4):201–210, 2021. https://doi.org/10.1049/smc2.12015

[8] S. Hoory, N. Linial, and A. Wigderson. Expander graphs and their applications. *Bulletin of the AMS*, 43(4):439–561, 2006. https://doi.org/10.1090/S0273-0979-06-01126-8

[9] C. Udokwu, R. Voicu-Dorobanţu, A. A. Ogunyemi, A. Norta, N. Sturua, and S. Craß. Leveraging blockchain for ethical AI: Mitigating digital threats and strengthening societal resilience. *Future Internet*, 17:309, 2025. https://doi.org/10.3390/fi17070309

[10] L.-H. Lee, T. Braud, P. Zhou, et al. All one needs to know about metaverse: A complete survey on technological singularity, virtual ecosystem, and research agenda. *arXiv preprint arXiv:2110.05352*, 2021. https://arxiv.org/abs/2110.05352

[11] G. Li, H. Hammoud, H. Itani, D. Khizbullin, and B. Ghanem. CAMEL: Communicative agents for "mind" exploration of large language model society. In *Advances in NeurIPS*, 2023. https://arxiv.org/abs/2303.17760

[12] A. Nechesov, I. Dorokhov, and J. Ruponen. Virtual cities: From digital twins to autonomous AI societies. *IEEE Access*, 13:13866–13903, 2025. https://doi.org/10.1109/ACCESS.2025.3531222

[13] L. Pan, A. Albalak, X. Wang, and W. Y. Wang. Logic-LM: Empowering large language models with symbolic solvers for faithful logical reasoning. In *Findings of EMNLP*, 2023. https://arxiv.org/abs/2305.12295

[14] M. S. Paterson and M. N. Wegman. Linear unification. *Journal of Computer and System Sciences*, 16(2):158–167, 1978. https://doi.org/10.1016/0022-0000(78)90043-0

[15] R. Riegel, A. Gray, F. Luus, et al. Logical neural networks. *arXiv preprint arXiv:2006.13155*, 2020. https://arxiv.org/abs/2006.13155

[16] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Pearson, 4th edition, 2022. https://aima.cs.berkeley.edu/

[17] Y. Shoham and K. Leyton-Brown. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge University Press, 2009. https://www.masfoundations.org/

[18] F. Tao, H. Zhang, A. Liu, and A. Y. C. Nee. Digital twin in industry: State-of-the-art. *IEEE Trans. Ind. Informat.*, 15(4):2405–2415, 2019. https://doi.org/10.1109/TII.2018.2873186

[19] A. Tarski. A lattice-theoretical fixpoint theorem and its applications. *Pacific Journal of Mathematics*, 5(2):285–309, 1955. https://projecteuclid.org/journals/pacific-journal-of-mathematics/volume-5/issue-2/A-lattice-theoretical-fixpoint-theorem-and-its-applications/pjm/1103044538.full

[20] H. Touvron, L. Martin, K. Stone, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023. https://arxiv.org/abs/2307.09288

[21] M. Wooldridge. *An Introduction to MultiAgent Systems*. Wiley, 2nd edition, 2009. https://www.cs.ox.ac.uk/people/michael.wooldridge/pubs/imas/IMAS2e.html

[22] Altera.AI. et al. Project Sid: Many-agent simulations toward AI civilization. *arXiv preprint arXiv:2411.00114*, 2024. https://arxiv.org/abs/2411.00114

[23] S. Goncharov and A. Nechesov. Polynomial-computable representation of neural networks in semantic programming. *J*, 6(1):48–57, 2023. https://doi.org/10.3390/j6010004

[24] Y. LeCun. A path towards autonomous machine intelligence. *OpenReview*, 2022. https://openreview.net/forum?id=BZ5a1r-kVsf

[25] H. Liu et al. Trustworthy AI: A computational perspective. *ACM Trans. Intell. Syst. Technol.*, 14(1):4, 2022. https://doi.org/10.1145/3546872

[26] N. F. Rajani et al. Explain yourself! Leveraging language models for commonsense reasoning. ArXiv. In *Proc. ACL*, 2019. https://arxiv.org/abs/1906.02361

[27] A. S. Abadi and L.-K. Soh. Challenges in credit assignment for multi-agent reinforcement learning in open agent systems. *arXiv preprint arXiv:2510.27659*, 2025. https://arxiv.org/abs/2510.27659

[28] W. Chen, Y. Su, J. Zuo, C. Yang, C. Yuan, et al. AgentVerse: Facilitating multi-agent collaboration and exploring emergent behaviors. *arXiv preprint arXiv:2308.10848*, 2023. https://arxiv.org/abs/2308.10848

[29] W. Chen, W. Li, X. Liu, S. Yang, and Y. Gao. Learning explicit credit assignment for cooperative multi-agent reinforcement learning via polarization policy gradient. *arXiv preprint arXiv:2210.05367*, 2023. https://arxiv.org/abs/2210.05367

[30] S. Han, Q. Zhang, W. Jin, and Z. Xu. LLM multi-agent systems: Challenges and open problems. *arXiv preprint arXiv:2402.03578*, 2024. https://arxiv.org/abs/2402.03578

[31] Z. Ning and L. Xie. A survey on multi-agent reinforcement learning and its application. *Journal of Automation and Intelligence*, 3(2):73–91, 2024. https://doi.org/10.1016/j.jai.2024.02.003

[32] J. S. Park, J. C. O'Brien, C. J. Cai, M. R. Morris, P. Liang, and M. S. Bernstein. Generative agents: Interactive simulacra of human behavior. *arXiv preprint arXiv:2304.03442*, 2023. https://arxiv.org/abs/2304.03442

[33] K.-T. Tran, D. Dao, M.-D. Nguyen, Q.-V. Pham, B. O'Sullivan, and H. D. Nguyen. Multi-agent collaboration mechanisms: A survey of LLMs. *arXiv preprint arXiv:2501.06322*, 2025. https://arxiv.org/abs/2501.06322

[34] S. Gronauer and K. Diepold. Multi-agent deep reinforcement learning: a survey. *Artificial Intelligence Review*, 55(2):895–943, 2022. https://doi.org/10.1007/s10462-021-09996-w

[35] S. Wu and K. Shu. Memory in LLM-based multi-agent systems: Mechanisms, challenges, and collective intelligence. 2025. https://doi.org/10.13140/RG.2.2.21084.04485

[36] S. Barykin, W. Zhang, D. Dinets, A. Nechesov, et al. Designing a Russian–Chinese omnichannel logistics network for the supply of bioethanol. *Sustainability*, 17:7968, 2025. https://doi.org/10.3390/su17177968

[37] A. Nechesov and S. Barykin. One AI, One World: A global AI strategy by the International AI Committee (IAIC). *System Informatics*, 26:77–102, 2025. https://doi.org/10.31144/si.2307-6410.2025.n26.p77-102