

Decoupling Correctness from Policy: A Deterministic Causal Structure for Multi-Agent Systems

Zhiyuan Ren, *Member, IEEE*, Tao Zhang, Wenchi Cheng, *Senior Member, IEEE*

Abstract—In distributed multi-agent systems, correctness is often entangled with operational policies such as scheduling, batching, or routing, which makes systems brittle since performance-driven policy evolution may break integrity guarantees. This paper introduces the Deterministic Causal Structure (DCS), a formal foundation that decouples correctness from policy. We develop a minimal axiomatic theory and prove four results: existence and uniqueness, policy-agnostic invariance, observational equivalence, and axiom minimality. These results show that DCS resolves causal ambiguities that value-centric convergence models such as CRDTs cannot address, and that removing any axiom collapses determinism into ambiguity. DCS thus emerges as a boundary principle of asynchronous computation, analogous to CAP and FLP: correctness is preserved only within the expressive power of a join-semilattice. All guarantees are established by axioms and proofs, with only minimal illustrative constructions included to aid intuition. This work establishes correctness as a fixed, policy-agnostic substrate—a "Correctness-as-a-Chassis" paradigm—on which distributed intelligent systems can be built modularly, safely, and evolvably.

Index Terms—Deterministic Causal Structure (DCS), Distributed systems, Multi-agent systems, Correctness, Formal methods, Causality

I. INTRODUCTION

A CORE TENSION in designing distributed multi-agent systems (MAS) lies in reconciling the goal of agent **autonomy** with the need for guaranteed system-wide **correctness**. This tension often arises from a deep-seated entanglement between an agent's operational **policy**, which dictates its autonomous behavior, and the system's underlying **structural correctness** (i.e., the integrity of its causal history). When an agent's behavioral logic is intertwined with the mechanisms that ensure historical integrity, the system becomes brittle: evolving an agent's policy to improve performance or adapt to new conditions risks violating global correctness guarantees. This paper confronts this core MAS challenge directly, focusing on the domain of collaborative, protocol-adherent MAS where the foundational problem is to establish deterministic correctness amidst asynchrony and policy diversity, rather than to defend against malicious, protocol-violating behavior.

This paper asks: can we achieve a formal **decoupling of correctness from policy**? We propose that a system's correctness should be anchored to a policy-agnostic *structural*

invariant—a mathematical object whose integrity is guaranteed regardless of the specific, and possibly non-deterministic, policies executed by the agents. Our answer is a constructive proof centered on a **Deterministic Causal Structure (DCS)**. We demonstrate that the complete causal history of agent interactions can be proven to converge to a unique, globally consistent *Provenance Directed Acyclic Graph (Provenance DAG)*. This DAG serves as the correctness invariant.

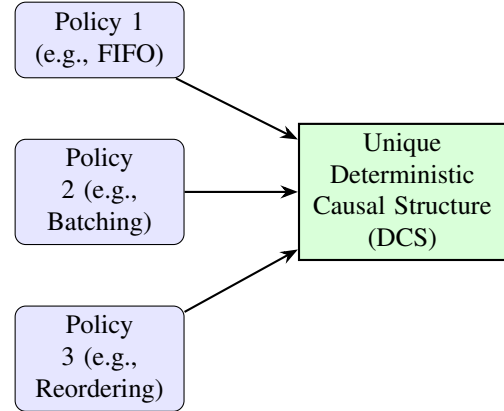


Fig. 1. The core principle of Structure-Policy Decoupling. This illustrates how, for the **exact same set of generated events**, different **operational policies** (such as message ordering, batching, or reordering) all result in the construction of the identical, unique Deterministic Causal Structure (DCS). The correctness of the causal history is thus decoupled from the infrastructural mechanics of its delivery.

It is crucial to define the scope of this decoupling precisely. The policy-agnosticism guaranteed by DCS separates the **Recording Mechanism** from the **Execution Path**. The system's final state, which is determined by the specific set of contributions generated over time (the Execution Path), remains fundamentally dependent on the agents' high-level generation policies. The guarantee of DCS is that for any given execution path, the infrastructural mechanism by which that history is recorded is deterministic and unique, regardless of the underlying operational policies (e.g., routing, scheduling) that delivered the information, as shown in Fig. 1.

To establish this, we present a complete theory that rests on three core propositions:

- 1) **Existence and Uniqueness of the DCS (Theorem 1):** We prove that under a minimal set of axioms, the global causal history graph (the DCS) exists, is unique up to isomorphism, and is constructively verifiable.

This work was supported by the National Key Research and Development Program of China (No. No.2023YFC3011502).

Zhiyuan Ren, Tao Zhang and Wenchi Cheng are with the School of Telecommunications Engineering, Xidian University, Xi'an 710071, China.

Corresponding author: Zhiyuan Ren (zyren@xidian.edu.cn).

- 2) **The Decoupling Invariant (Theorem 2):** We prove that this DCS is a policy-agnostic invariant. For any two admissible policies (e.g., different schedulers or routing strategies), the resulting DCS graphs are guaranteed to be isomorphic. This formally separates the system’s correctness from its operational policies.
- 3) **Observational Equivalence (Proposition 1):** We prove that for a broad class of applications, two executions are observationally indistinguishable *if and only if* their DCS graphs are isomorphic. This establishes the DCS as the minimal and sufficient carrier of information for correctness.

This decoupling provides a new paradigm for system design. Instead of reasoning about complex, holistic system behaviors, developers can first establish the correctness of the structural invariant, and then safely and independently explore the vast space of performance-optimizing policies. Our core contributions are:

- 1) We formalize the structure-policy entanglement problem and propose its decoupling as a core design principle for robust distributed systems.
- 2) We present the theory of a Deterministic Causal Structure (DCS) and prove it serves as a policy-agnostic invariant for correctness (Theorems 1 & 2).
- 3) We complete the theory by proving that DCS isomorphism is a necessary and sufficient condition for observational equivalence for a wide range of applications (Proposition 1).
- 4) We formally prove that the structural guarantee of a DCS is strictly stronger than the value convergence of models like CRDTs.

The remainder of this paper is organized as follows. Section II reviews related work to position our contribution. Section III establishes our formal model and core axioms. Section IV presents the core theoretical results of the DCS theory. Section V provides a set of illustrative constructions to intuitively demonstrate the implications of our theoretical results. Section VI discusses the broader implications and theoretical boundaries of our work. Finally, Section VII concludes the paper.

II. RELATED WORK

The DCS theory is situated at the intersection of research in distributed systems and multi-agent systems. To clearly position the uniqueness and contributions of our work, this section provides a systematic comparison of our theory against several key related research areas. We will sequentially discuss: classical distributed consensus protocols; replicated data types and eventual consistency models; causal consistency theories; DAG-based BFT consensus; and data provenance. Through these comparisons, we will elucidate the fundamental innovation of DCS in guaranteeing structural determinism.

A. Distributed Consensus and Total Order Broadcast

Classical consensus protocols, foundational to distributed systems, aim to solve the State Machine Replication (SMR)

problem by enforcing a **Total Order Broadcast** [1]. Early practical solutions like Viewstamped Replication [2], and later the widely-known Paxos [3] and Raft [4], provided mechanisms to achieve this. However, the performance limitations of this strictly sequential execution on modern multi-core architectures are well-documented [5], [6]. Consequently, a significant body of recent research has focused on overcoming this bottleneck, through avenues like *parallel state machine replication* [5], [6], hybrid systems that switch between deterministic and optimistic execution [7], and state partitioning to enable scalability [8].

However, a common thread unites these protocols: they achieve correctness by definitionally **entangling it with a specific ordering policy**. To linearize naturally concurrent events, the system must employ a policy (e.g., a leader’s sequencing decision) to dictate an arbitrary order between them [8]. The final log—the carrier of correctness—is therefore a direct artifact of this policy. The DCS theory offers a fundamentally different approach based on decoupling. Instead of enforcing a policy-dependent total order, it seeks consensus on the objective, *policy-independent Causal Partial Order*. The DCS preserves the natural concurrency of interactions, capturing the "ground truth" of what happened without imposing an artificial timeline, thus providing a stable structural invariant upon which diverse performance strategies can be independently optimized.

B. Replicated Data Types and Value Convergence

Conflict-free Replicated Data Types (CRDTs) [9], [10] are philosophically the closest to our work, as they also embrace concurrency. While earlier techniques like Operational Transformation (OT) [11] addressed concurrent updates, they often required central coordination, a challenge that persists in modern implementations [12]. CRDTs, grounded in theoretical principles like the CALM theorem [13], instead guarantee that replicas converge to the same final **value** in a decentralized manner, regardless of message delivery order. This powerful form of **Value Convergence** has seen significant optimization, such as Delta State CRDTs which reduce network overhead by propagating only incremental changes [14].

However, this guarantee of value convergence is achieved by wedding correctness to a policy of *causal indifference*. The semilattice merge function is, by design, insensitive to the causal relationship between operations; it only cares about the set of operations to be merged. This focus on value at the expense of causality has been a subject of further research [15], [16], and has led some systems to use a partially-ordered log, such as a DAG, as the underlying structure for the historical record, while relying on CRDTs at the application layer to interpret the state from this log [17], [18]. This leads to what we have termed *Structural Ambiguity*: as formally proven in Proposition 2 and illustrated in Fig. 8, two executions with non-isomorphic causal histories can produce the same final value. The DCS theory provides a strictly stronger guarantee by establishing a policy-agnostic **structural invariant**, resolving the structural ambiguity that CRDTs ignore.

C. Causal Consistency and Logical Clocks

Causal consistency, often implemented with tools like Lamport Clocks [19] and Vector Clocks [20], [21], ensures that events are processed in an order that respects causality. This model is often offered as an intermediate guarantee between strong consistency and eventual consistency [22], ensuring that causally related operations are seen by all replicas in the same order. This model found significant practical application in large-scale systems like Amazon’s Dynamo, which used vector clocks to manage versioning in a highly available key-value store [23].

However, this guarantee is fundamentally a **local reasoning tool**, not a global, convergent invariant. While it enforces a correct *local processing policy*, it does not guarantee that the resulting global history graph will be unique across all nodes. Since concurrent, causally independent operations are not ordered with respect to each other, different replicas may process them in different sequences, resulting in divergent and non-isomorphic views of the history that are both, from their local perspectives, causally consistent [22]. DCS solves this by shifting the burden of correctness from the local processing policy to the **data structure itself**. By enforcing immutable, universally consistent metadata (*rid*, *parents*), the DCS theory guarantees the emergence of a single, global structural invariant, decoupling the system’s ground truth from the partial, policy-dependent views of individual agents.

D. DAG-based BFT Consensus

Recent high-performance BFT consensus protocols leverage a Directed Acyclic Graph (DAG) structure to decouple data dissemination from ordering. Systems like Hedera Hashgraph [24], Aleph [25], Narwhal and Tusk [26], and Bullshark [27] have demonstrated significant performance gains using this approach. In these systems, nodes first form a DAG of transactions, which efficiently records the causal partial order. However, in all these designs, the DAG is merely a **means to an end**: the ultimate goal remains to establish a **Total Order**. This is evident in recent research, where DAGs are used to enhance existing protocols but must still be resolved into a final, linear chain [28], and where significant effort is focused on optimizing the secondary **ordering policy** itself, using everything from lightweight broadcast primitives [29], [30] and theoretical quorum analysis [31] to learning-based approaches with GNNs [32].

Correctness is thus defined by the output of this policy-dependent linearization process. In sharp contrast, the DCS theory posits that the unique, globally consistent DAG is **the end itself**. We provide a **structural consensus**, not an ordering consensus. The goal is not to flatten the rich partial order into a policy-dependent total order, but to agree on the policy-*independent* structure of the partial order itself.

E. Data Provenance and Scientific Workflows

Our work is deeply connected to Data Provenance, as the DCS is, by definition, a Provenance DAG. The field of data provenance, codified in standards like the W3C PROV model,

studies the origin and history of data, with significant research on representing and querying these graphs [33]. However, traditional provenance systems often focus on representation, implicitly **entangling the integrity of the graph with a specific recording policy**, such as a trusted, centralized logger. This assumption breaks down in decentralized environments, leading recent research to turn to distributed ledger technology (DLT) to construct a secure and tamper-evident data lineage [34], [35], sometimes requiring novel consensus mechanisms tailored for provenance to ensure the history is trustworthy [36].

The DCS theory addresses this same fundamental problem—how to construct this graph in a decentralized, trustless environment—but from a different perspective. Instead of relying on an external, application-level consensus protocol or a generic DLT, it achieves the **decoupling of the graph’s integrity from any recording policy** through its axiomatic foundation. Correctness is not guaranteed by an external mechanism layered on top, but by the axiomatic, immutable properties of the data contributions themselves. The DCS is therefore not a new model for representing provenance, but a *consensus protocol for constructing a policy-agnostic, verifiable Provenance DAG*. It provides the missing formal foundation for building truly decentralized and auditable scientific workflows.

F. Blockchain and Distributed Ledger Technologies

Classical blockchains, exemplified by Bitcoin [37] and Ethereum [38], establish an immutable ledger by achieving a probabilistic **Total Order Consensus**. This model’s correctness is inextricably **entangled with a resource-intensive admission policy**, like Proof-of-Work (PoW) or Proof-of-Stake (PoS) [39], which creates a bottleneck. While DAG-based blockchains were developed to overcome these limitations by allowing concurrent block creation [40], [41], simply adopting a DAG structure does not eliminate the need for higher-level policies. Advanced systems like MorphDAG and RT-DAG, for instance, superimpose complex workload-aware or real-time ordering policies on the DAG to manage concurrency and transaction patterns [42], [43].

The DCS paradigm offers a fundamentally different vision for a distributed ledger. It not only decouples the ledger’s integrity from a resource-intensive *admission policy* like PoW, but also from the complex *workload-aware or real-time ordering policies* required by advanced DAG blockchains. Instead of a linear chain, the ledger *is* the unique Provenance DAG itself. Correctness is not defined by a single, policy-ordered history, but by the policy-agnostic, intrinsic structure of all interactions. This suggests a new type of distributed ledger technology where the primary goal is not total ordering, but consensus on the complete, causally-rich, and policy-independent graph of the interaction history.

In summary, this review reveals that existing paradigms consistently entangle system correctness with operational policy, a conceptual overview of which is presented in Table I. This pervasive entanglement motivates our work to formally decouple structural integrity from policy, which the following sections now develop.

TABLE I
CONCEPTUAL COMPARISON OF DCS WITH RELATED DISTRIBUTED SYSTEMS PARADIGMS

Paradigm	Core Goal	Object of Consensus	Structure	Policy Dependence
DCS (Our Work)	Structural Correctness (Decoupled from Policy)	Causal Partial Order (The unique DAG itself)	Deterministic Causal DAG	Policy-Agnostic
Classical Consensus (e.g., Paxos, Raft)	State Machine Replication (SMR)	Total Order Broadcast	Linear Log / Chain	Tightly Entangled (e.g., Leader's sequence)
Replicated Data Types (CRDTs)	Value Convergence	Final State Value (Merge Function)	Structurally Ambiguous (History is lost)	Entangled with Causal Indifference
Causal Consistency (e.g., Vector Clocks)	Local Causal Ordering	None (Local validation, no global agreement)	Divergent Local Views (Non-isomorphic)	Entangled with Local Reception Order
DAG-based BFT (e.g., Narwhal, Tusk)	High-Throughput SMR	Total Order (Extracted from DAG)	Intermediate DAG, Final Linear Log	Tightly Entangled (Secondary ordering protocol)
Data Provenance & Blockchain	Verifiable History / Asset Transfer	Total Order (Blockchain) / None (Traditional Provenance)	Linear Chain / Centrally Recorded Graph	Entangled with Recording or Admission Policy (e.g., PoW)

III. THE FORMAL FRAMEWORK: MODEL AND AXIOMS

To deduce the intrinsic structural laws of multi-agent interactions from first principles, we must first construct a precise formal framework. This section establishes this framework by first defining our system model and fault assumptions, then introducing a minimal interaction model, and finally proposing a set of core axioms. These axioms represent the minimal set of formal constraints required to guarantee the emergence of a Deterministic Causal Structure.

A. System Model and Fault Assumptions

We consider a standard distributed system composed of a set of agents V . The system operates under the following assumptions:

- **Asynchronous Model:** The system is asynchronous. There is no global clock, and we make no assumptions about the relative processing speeds of agents or message delivery times, other than that they are finite.
- **Non-Byzantine Fault Model:** Agents are assumed to be "honest-but-fallible." They correctly follow the protocol rules (i.e., the axioms) at all times, but may fail at any time by crashing (the crash-stop model). This is a deliberate and fitting model for collaborative MAS, where agents are code-driven and operate in a permissioned environment. Unlike open, adversarial systems, the fundamental problem here is not malice, but ensuring resilience and deterministic outcomes despite software faults and network uncertainty. We therefore do not consider arbitrary or protocol-violating (Byzantine) behavior.
- **Unreliable Network Model:** The network connecting the agents is unreliable. It may lose, duplicate, or reorder messages at will. The only liveness guarantee on communication is a weak fairness property, which will be formally stated in Axiom 1.

B. Interaction Model

The model aims to capture the core process of agents building a shared body of knowledge through atomic contributions, given the system model defined above.

a) Basic Components: The system consists of a finite **Set of Agents**, denoted as $V = \{1, \dots, n\}$, and an abstract **Key Space**, denoted as K , used to logically partition interactions.

b) State and Merging: For any agent $i \in V$ and key $k \in K$, its **Local State** is denoted as $M_i(k)$. Each key's state belongs to a **State Space** $(L_k, \sqsubseteq, \sqcup)$, which we require to be a **directed-complete join-semilattice** (dcpo-join-semilattice). A **join-semilattice** is a partially ordered set where the join operation $\sqcup$ is associative, commutative, and idempotent.

c) The Canonical Representation of Interaction: The **Contribution**: All interactions in our model are captured by a single, canonical data structure called a **Contribution**, denoted by δ . We formally define it as a tuple:

Definition 1 (Contribution). A contribution δ is a tuple $(rid, parents, payload, k)$, where:

- $rid \in \mathcal{R}$ is a globally unique identifier for the contribution.
- $parents$ is a finite set of $rids$, $\{p_1, p_2, \dots, p_m\}$, representing the direct causal predecessors of this contribution.
- $payload$ contains the arbitrary, application-specific data.
- $k \in K$ is a key used for logical partitioning of contributions.

This specific tuple structure is not arbitrary, but is itself a minimal requirement for achieving a DCS. The `rid` provides a unique identity for each event. The `parents` set is essential for explicitly encoding the causal dependencies that form the very fabric of the history graph. The separation of `payload` from the immutable metadata $(rid, parents)$ is the key to enabling policy-agnosticism, as our axioms will only constrain the metadata, leaving the content entirely free. Therefore, this data format is the necessary substrate upon which our axioms operate to guarantee structural convergence.

d) System Execution Model: We define the **Universe of Contributions** as the set $\mathcal{R}$ of all possible `rids`. For any key $k \in K$, a **Relevant Agent Set**, $Rel(k) \subseteq V$, represents the members that subscribe to information for this key. For any agent $i \in Rel(k)$, its **Final Mergeable Set**, $Merge_i(k) \subseteq \mathcal{R}$, is the set of `rids` it is guaranteed to eventually receive and merge.

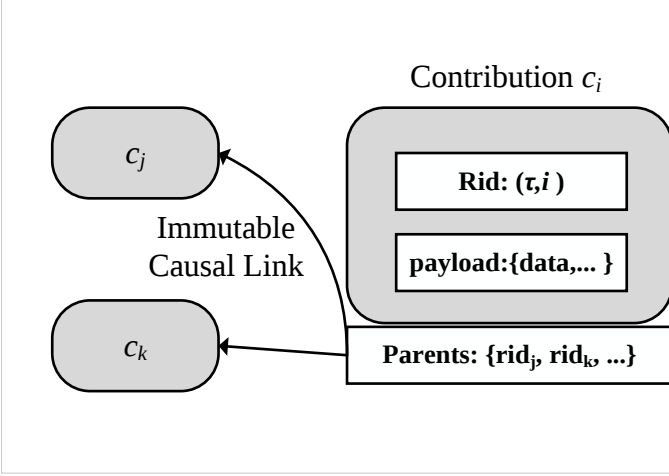


Fig. 2. The internal structure of a Contribution. The immutable causal history is "soldered" into the data structure itself via the `parents` field, which contains the unique RIDs of its direct predecessors. This forms an unbreakable, verifiable causal chain.

C. Core Axioms

Having defined the components of our system, we now establish the fundamental rules governing their interaction. The following axioms are not arbitrary postulates, but rather the logical consequences derived from the requirements for achieving a Deterministic Causal Structure in an asynchronous, decentralized environment. We will present them constructively by addressing the key challenges of communication, state management, and historical integrity.

a) *The Communication Postulate*: The first foundational challenge is communication. In an asynchronous, unreliable network, messages can be lost. If any agent is permanently unable to see a piece of the interaction history that others have seen, then no global convergence of any kind is possible. Therefore, any viable theory must begin with a baseline liveness guarantee. We do not need to assume a strong property like reliable or ordered delivery, but we must posit a minimal fairness condition that ensures all relevant information is eventually propagated. This leads to our first axiom.

Axiom 1. (Localized Weak Fairness) For any key $k \in K$, if a contribution δ belonging to that key is persistently sent, then δ will eventually be delivered at least once to all agents in the relevant set $\text{Rel}(k)$.

b) *The Algebraic Foundation for State*: The second challenge stems from the consequences of asynchronicity: message reordering and duplication. If the result of integrating new information into an agent's local state depends on the order of arrival, the system's outcome will be non-deterministic. To eliminate this ambiguity, the state update mechanism must be inherently order-agnostic. This requires the state space to possess specific algebraic properties—namely, that the merge operation is associative, commutative, and idempotent. A join-semilattice is the formal structure that precisely captures these properties. The requirement of being "directed-complete" is a further technical condition to rigorously handle the limits of infinite executions.

Axiom 2. (Directed-Complete Join Semilattice) For any key $k \in K$, its state space $(L_k, \sqsubseteq, \sqcup)$ must form a directed-complete join-semilattice.

c) *The Integrity of Historical Facts*: Finally, and most critically, even with guaranteed communication and an order-agnostic state model, a DCS is impossible if the historical records themselves—the contributions—are not well-behaved. To construct a single, unambiguous history, each event or "fact" within that history must be immutable and causally well-formed. We enforce this through a set of integrity axioms. First, each fact must have a unique identity and unchangeable content.

Axiom 3. (Contribution as an Immutable Fact) Every contribution δ represents a self-contained, immutable historical fact. To guarantee this, its identity, the `rid`, must be globally unique, and its substance, the `payload`, must be immutable upon creation.

Axiom 4. (Immutability of Causal Linkage) To construct a single, unambiguous causal graph, the relational links between contributions must be permanently fixed. Therefore, the `parents` set of every contribution δ , which encodes its direct causal dependencies, must be immutable upon creation.

Second, beyond immutability, each fact must respect the arrow of time. An event cannot be caused by something that has not yet happened. This principle of causal well-formedness is essential to prevent paradoxes and guarantee an acyclic history.

Axiom 5. (Causal Well-Formedness) When an agent generates a new contribution δ , all `rids` in its `parents` set must be drawn from the set of contributions already observed by that agent at the time of creation.

IV. CORE THEORETICAL RESULTS

Having established the formal framework and the core problem of policy-correctness entanglement, this section presents our solution: the formal theory of a **Deterministic Causal Structure (DCS)** as a policy-agnostic invariant. We will first prove the existence and uniqueness of the DCS (Theorem 1), then prove its invariance under any admissible policy (Theorem 2), and finally, formally connect this structural isomorphism to the concept of observational equivalence (Proposition 1), thus completing our decoupling argument.

A. Theorem A: Existence, Uniqueness, and Constructibility of the DCS

We begin by proving that in any system adhering to our axioms, the seemingly chaotic interactions will inevitably converge to a single, well-defined mathematical object.

Theorem 1 (Existence, Uniqueness, and Constructibility of the DCS). Let the global interaction graph be a **Provenance DAG**, denoted by $G^* = (\mathcal{R}, E)$, where $\mathcal{R}$ is the set of all contribution `rids`, and $E = \{(p \rightarrow r) \mid p \in \text{parents}(r)\}$ is the set of causal dependency edges. Under Axioms 1 through 5:

- 1) (**Existence**) The global interaction graph G^* is a well-defined Directed Acyclic Graph.
- 2) (**Uniqueness**) The structure of G^* (up to graph isomorphism) is unique, independent of the order of contribution generation and propagation.
- 3) (**Constructibility**) For any key k and any agent $i \in \text{Rel}(k)$, its local view of the graph converges, and the limit of its local state $M_i^*(k)$ exists, is unique, and is equal to the join of all payloads in its final mergeable set.

Proof Sketch: The proof demonstrates how distinct subsets of our axioms guarantee each property of the DCS. (1) **Existence** of a well-defined DAG is guaranteed primarily by Axiom 5 (Causal Well-Formedness), which prohibits cycles. (2) **Uniqueness** of this structure up to isomorphism is a direct consequence of Axioms 3 and 4, which enforce the immutability of the graph’s metadata (`rid` and `parents`). Finally, (3) **Constructibility** of a deterministic state from this unique structure is made possible by Axioms 1 and 2, which ensure that all information is eventually received and can be merged in a policy-agnostic, order-independent manner. The full formal proof is detailed in Appendix B. ■

B. Theorem B: The Decoupling Invariant — Policy-Agnosticism

Theorem A establishes the DCS as a stable, unique structure. The next theorem reveals its most powerful property: this structural integrity is completely independent of the agents’ behavior, providing the formal basis for our decoupling claim.

Theorem 2 (The Decoupling Invariant: Policy-Agnosticism). *For any two admissible agent policies P_1 and P_2 , which may differ in any aspect of their operation (e.g., scheduling, batching, routing), if they generate the same set of contributions, the resulting global DCS graphs G_1^* and G_2^* are isomorphic.*

Proof Sketch: The proof follows directly from the logic of Theorem 1. A review of that proof reveals that its every step relies solely on the formal properties of the contribution’s metadata (`rid`, `parents`), as constrained by our axioms. The proof logic never inspects the `payload` content, nor does it impose any constraints on the timing, frequency, or ordering of contribution generation, which constitute the agent’s policy. Therefore, the structural conclusion of Theorem 1 is necessarily independent of any specific policy, establishing the DCS as a policy-agnostic invariant. The full proof is detailed in Appendix B. ■

C. Proposition C: Observational Equivalence

Theorems A and B show we can decouple a structural invariant from policy. But why is this specific invariant the *right* one? This final proposition closes the loop by proving that the DCS structure is precisely the information needed for correctness: two systems are indistinguishable from the outside if and only if their internal structures are the same.

Proposition 1 (Observational Equivalence). *For all upper-layer computations that use only (i) ancestor and concurrency*

queries on the DCS and (ii) semilattice-homomorphic aggregates over payloads, two executions are observationally indistinguishable if and only if their DCS graphs are isomorphic.

Proof Sketch: The proof proceeds in two directions. ($\Leftarrow$) If the DCS graphs are isomorphic, any structural query (like `is_ancestor(p,r)`) will yield identical results. Since payloads are immutable and uniquely identified by `rids`, any semilattice-based aggregation over them will also be identical. Thus, the executions are observationally indistinguishable. ($\Rightarrow$) If the DCS graphs are non-isomorphic, there must exist a structural difference (e.g., an edge $(p \rightarrow r)$ exists in one but not the other). We can then construct a simple structural query that distinguishes the two executions. This proves that the DCS contains the minimal and sufficient information for this class of applications. The full formal proof is detailed in Appendix C. ■

This completes the core of our decoupling theory. The following subsections further solidify its foundations by proving the necessity of our axioms and formally positioning it against existing models.

D. The Theoretical Boundary: Minimality of the Axioms

The preceding theorems have demonstrated the power of the DCS theory—guaranteeing structural determinism even for fully autonomous “black box” agents. A natural and rigorous question to ask is whether we have introduced redundant or overly strong axioms to achieve this powerful guarantee, thereby unnecessarily narrowing the theory’s scope of applicability. A complete theory must demonstrate not only the sufficiency of its premises, but also their necessity. The following theorem addresses this question directly by proving the minimality of our axiom set, thereby establishing the sharp and solid boundaries of our theory.

Theorem 3 (Axiom Minimality). *The axiom set {Axiom 1, 2, 3, 4, 5} is the minimal set required to guarantee a DCS. If any one of these axioms is removed, a constructive counterexample exists where the guarantees of Theorem 1 fail.*

Proof Sketch: The sufficiency of this axiom set is established by the proof of Theorem 1. The necessity is proven by constructing counterexamples that demonstrate how removing axioms from each subset shatters a distinct aspect of the DCS guarantee. As detailed in Appendix D, removing axioms from the **Structural Integrity** set (Axioms 3-5) leads to non-unique or ill-defined causal graphs (Fig. 5, 6, 7), while removing axioms from the **Deterministic State Interpretation** set (Axioms 1-2) makes it impossible to derive a consistent, policy-agnostic state from the graph, even if the structure itself were unique (Fig. 3, 4).

- (i) **Removing Axiom 1 (Weak Fairness):** This allows for network partitions where some agents never receive critical contributions, leading to permanently divergent final states. This failure of value convergence is illustrated in Fig. 3.
- (ii) **Removing Axiom 2 (Join-Semilattice):** Without the algebraic properties of a join-semilattice, the merge

operation may not be commutative or idempotent. This makes the final state dependent on the arbitrary arrival order of messages, as shown in Fig. 4.

- (iii) **Removing Axiom 3 (Contribution as an Immutable Fact):** If contribution metadata like the `rid` is not unique, the global graph becomes ill-defined, as different events may claim the same identity. This is illustrated in Fig. 5.
- (iv) **Removing Axiom 4 (Immutability of Causal Linkage):** If contribution metadata is not immutable, different executions can result in non-isomorphic but equally valid causal graphs for the same set of events, destroying structural uniqueness, as depicted in Fig. 6.
- (v) **Removing Axiom 5 (Causal Well-Formedness):** This permits the formation of forward-references, allowing contributions to form cycles. The global history is consequently no longer a DAG, and the foundational premise of the theory is broken, as shown in Fig. 7.

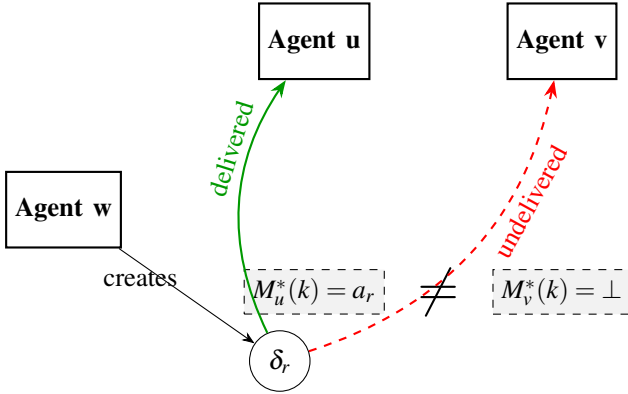


Fig. 3. Violation of Axiom 1 (Weak Fairness). The failure to deliver a contribution to all relevant agents results in permanently divergent local states ($M_u^* \neq M_v^*$), breaking the guarantee of value convergence.

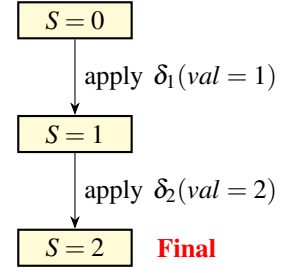
The proof of axiom minimality provides the final structural reinforcement for our theoretical edifice. It demonstrates that our proposed axiom set is precisely the necessary set of “physical laws” required to guarantee a DCS. At this point, we have fully defined our theory, proven its powerful internal properties, and established the rigor of its foundations. The final logical step, therefore, is to formally position this complete theory against the state-of-the-art models in the field of distributed consistency. The next proposition will, through a formal comparison, clearly reveal the essential differences and superiority of a DCS over models such as CRDTs.

E. The Formal Positioning: Superiority over Value Convergence

Finally, to precisely locate our contribution within the landscape of distributed consistency models, we formally prove that the structural guarantee of a DCS is strictly stronger than the value convergence offered by models such as CRDTs.

Proposition 2 (Separation from CRDTs). *There exists a system that satisfies the standard CRDT conditions (a join-semilattice state, an associative, commutative, and idempotent*

(a) **Schedule A:** δ_1 then δ_2



(b) **Schedule B:** δ_2 then δ_1

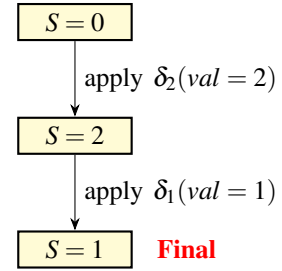


Fig. 4. Violation of Axiom 2 (Join-Semilattice). Without a commutative and idempotent merge operation (e.g., using an “overwrite” logic), the final state becomes dependent on the arbitrary message arrival order, thus violating determinism.

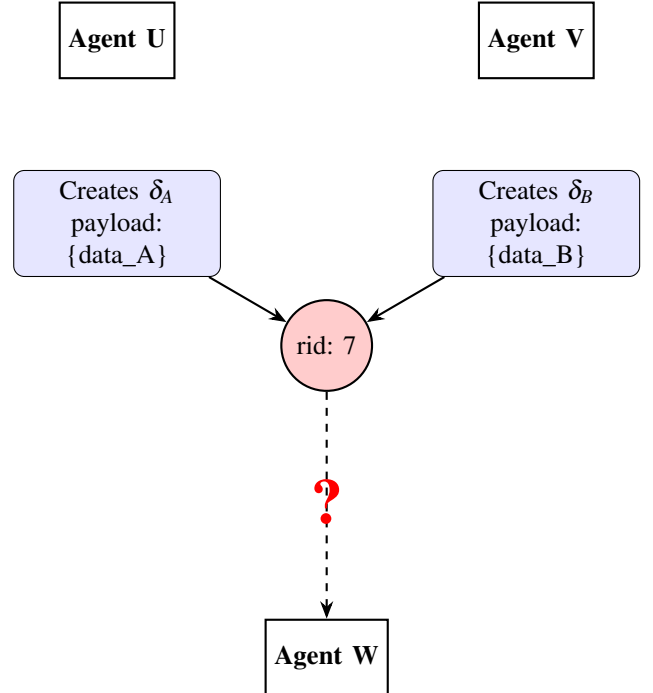


Fig. 5. Violation of Axiom 3 (Contribution as an Immutable Fact). If the uniqueness of a `rid` is not guaranteed, two different contributions (δ_A and δ_B) can claim the same identity. This makes the global graph ill-defined and violates structural determinism.

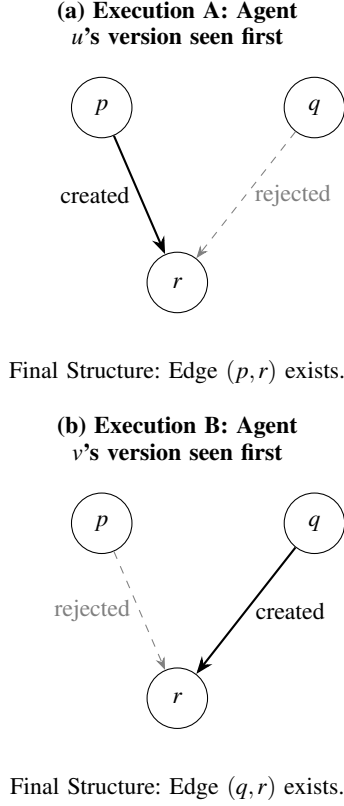


Fig. 6. Violation of **Axiom 4 (Immutability of Causal Linkage)**. If the parents metadata is mutable, two valid executions can produce non-isomorphic causal graphs. This is a direct illustration of the *Structural Ambiguity* problem, which violates the guarantee of a unique DCS.

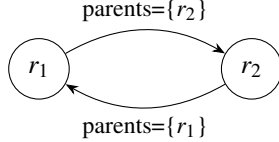


Fig. 7. Violation of **Axiom 5 (Causal Well-Formedness)**. The prohibition of forward-references is essential. Its removal allows for the formation of cycles, breaking the foundational Directed Acyclic Graph (DAG) property required for a DCS.

merge function, and fair communication) where the state value is guaranteed to converge, yet two valid executions can produce non-isomorphic causal histories.

Proof Sketch: The proof relies on a simple constructive example, as illustrated in Fig. 8. Two contributions can be generated concurrently in one execution and sequentially (causally) in another. Both executions yield the same final state value (e.g., the set union of their payloads), satisfying CRDT guarantees. However, their underlying causal structures are fundamentally different. A DCS, through its enforcement of immutable causal metadata (Axioms 3 and 5), distinguishes between these two histories, whereas a value-centric model cannot. This demonstrates the formal separation and superiority of our structural guarantee. ■

This final proposition concludes the formal presentation of our core theory. In this section, we have built our argument

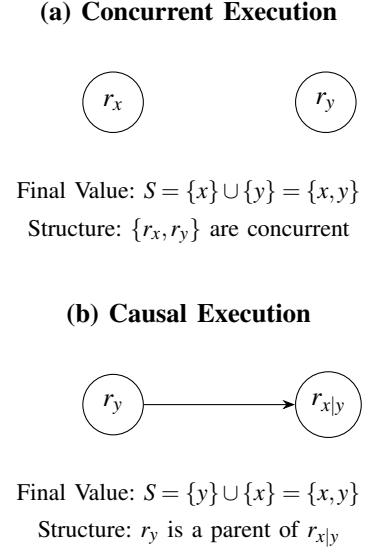


Fig. 8. Minimal construction demonstrating the separation of the DCS guarantee from CRDTs, arranged vertically for clarity. Both executions yield the same final value, satisfying CRDT guarantees. However, their causal structures are non-isomorphic, a distinction captured by a DCS but not by value-centric models.

from the ground up: we first proved the existence of a Deterministic Causal Structure and revealed its critical policy-agnostic property; we then reinforced the theory's foundations by proving the minimality of its axioms; and finally, we established its novelty and superiority through a formal comparison with value-convergence models. A complete and self-consistent theoretical system for structural convergence is now established. The following sections will step back from the formal proofs to discuss the broader implications and practical considerations of this new theory.

V. ILLUSTRATIVE CONSTRUCTIONS

The core theoretical results of this paper have been formally established in Section IV. To aid intuition, this section presents a set of *illustrative constructions* designed to visually and quantitatively demonstrate the implications of our theorems. These examples are **not performance benchmarks**, but rather serve to reinforce the correctness and necessity of our formal theory.

A. Axiom Necessity

To illustrate Theorem 3 (Axiom Minimality), we compare a fully DCS-compliant system against variants where key integrity axioms are violated. For each case, we execute the simulation twice from an identical initial state and test for structural isomorphism in the resulting histories. A non-zero ambiguity rate indicates a failure of determinism.

The results starkly demonstrate that removing any core integrity axiom collapses determinism into total ambiguity, visually confirming that the axiom set is indeed minimal and necessary.

TABLE II
DEMONSTRATION OF STRUCTURAL AMBIGUITY UPON AXIOM VIOLATION

System Type	Violated Axiom(s)	Ambiguity Rate
DCS (Baseline)	<i>None</i>	0%
Non-DCS Control	Metadata Mutability (Axiom 4)	100%
Non-DCS Control	Causal Forgery (Axiom 5)	100%

B. Policy-Agnosticism

To illustrate Theorem 2 (The Decoupling Invariant), we compare three heterogeneous policies: a static optimal routing strategy, a reinforcement-learning Q-routing agent, and an adaptive variant. Despite having radically different internal mechanics, for the subset of tasks where all three policies produced the exact same final path, their resulting Provenance DAGs were found to be isomorphic in **100%** of cases. This confirms that the DCS invariant correctly captures *what happened* independent of *how it happened*, thereby formally decoupling correctness from policy.

C. Distinctiveness of Policies

Finally, to validate that the policies being compared are genuinely heterogeneous, we briefly present their distinct performance characteristics.

TABLE III
PERFORMANCE CHARACTERISTICS OF HETEROGENEOUS POLICIES

Policy	Mean Hops	Std. Dev.	Success (%)
Static Optimal	4.02	1.24	100%
Q-Routing	4.29	1.35	100%
Adaptive Q-Routing	4.38	1.50	100%

While the causal structures they produce are identical when their execution paths align, their performance characteristics remain quantifiably different. This demonstrates that the DCS invariant preserves correctness without erasing meaningful diversity in policy efficiency.

In summary, these minimal constructions visually confirm the necessity of the axioms and the policy-agnostic nature of the DCS. All formal guarantees, however, are already proven in Section IV; this section merely serves to illustrate those guarantees in action.

VI. DISCUSSION AND PRACTICAL CONSIDERATIONS

The preceding section established the formal theory of the DCS as a policy-agnostic invariant. This section steps back from the formal proofs to explore the broader significance of this result, arguing that the **decoupling of correctness from policy** is not merely a theoretical curiosity, but a powerful new paradigm for designing and evolving complex distributed systems.

A. A New Design Paradigm: *Correctness-as-a-Chassis*

The core implication of our theory is a paradigm shift in system design. Traditionally, correctness is a holistic emerging

property of the entire system, including its policies. Our work reframes correctness as a fixed, verifiable "**chassis**", the DCS, upon which any compatible operational "**engine**", the policy, can be mounted. This **Correctness-as-a-Chassis** model, illustrated in Fig. 9, has profound theoretical and practical implications.

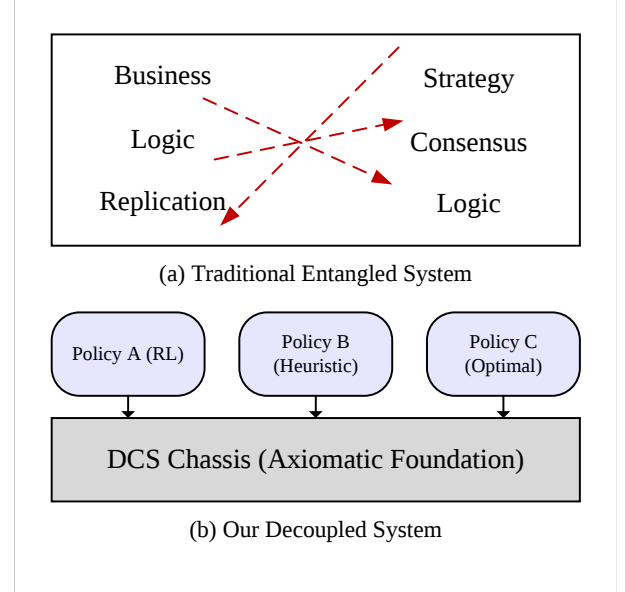


Fig. 9. Architectural comparison between traditional systems and our DCS framework. (a) In traditional systems, policy logic is deeply entangled with correctness logic. (b) Our framework provides a **Correctness-as-a-Chassis**, where an axiomatic foundation allows diverse policies to be plugged in interchangeably without compromising structural integrity.

a) Safe and Independent Evolution: The single greatest benefit of this decoupling is enabling the safe and independent evolution of a system's performance and its logic. Engineering teams can aggressively optimize policies (e.g., implementing a new batching strategy for higher throughput or a speculative execution model for lower latency) without fear of corrupting the system's structural integrity. As long as the new policy respects the axioms at the contribution level, the correctness of the overall system, as captured by the DCS, is guaranteed. This dramatically reduces the complexity of system evolution and enables continuous, safe refactoring.

b) Formal Verifiability and Auditing: The DCS transforms the system's interaction history from an opaque, policy-dependent artifact into a deterministic "white box". Because the final Provenance DAG is unique and constructible, it serves as an immutable ground truth for the system's entire causal history. This provides a solid foundation for high-stakes applications requiring formal verification, auditing, and replay, such as in decentralized finance or safety-critical autonomous systems. The audit is performed on the structure, which is guaranteed to be independent of any specific policy that was running at the time.

c) Composable Systems: Decoupling provides a clean contract for system composition. Two or more systems, each built upon a DCS foundation, can be composed with a much higher degree of confidence. The interaction between them can itself be modeled as a set of contributions, resulting

in a higher-level DCS. This allows for modular reasoning, where the correctness of each component can be analyzed independently of the others' internal policies, paving the way for a true marketplace of verifiable, interoperable intelligent components.

B. System Design Implications: The Power of Decoupling

The "Correctness-as-a-Chassis" paradigm has direct and powerful implications for system design, simplifying implementation and unlocking performance optimization.

a) *Implementation Simplicity and Emergent Consensus:* Implementing the core guarantees of the DCS theory is remarkably lightweight. Because correctness is guaranteed by the immutable, axiomatic structure of the data itself, the system does not require complex, state-based consensus protocols like Paxos or Raft. A minimal implementation consists of only three simple parts: 1) a data structure for contributions with immutable metadata (`rid`, `parents`); 2) any communication protocol, such as standard gossip, that satisfies weak fairness (Axiom 1); and 3) a local key-value store. In such a system, consensus is not actively "negotiated" through multi-phase commits; it is an Emergent Consensus—a property that is guaranteed to arise naturally from the structure of the data itself.

However, it must be acknowledged that while the consensus logic is lightweight, maintaining the complete Provenance DAG introduces overhead in storage and communication (e.g., the 'parents' metadata). This is the necessary trade-off for achieving the stronger guarantee of structural determinism over mere value convergence. Managing this overhead through techniques like graph pruning or garbage collection is therefore a key area for engineering optimization.

b) *Performance as an Independent Optimization Layer:* This minimal implementation prioritizes correctness and simplicity, not raw performance. However, the true power of the decoupling paradigm is that it treats **performance as an independent, pluggable optimization layer**. The DCS axioms provide the rigid "correctness chassis." Upon this foundation, engineers are free to design, test, and deploy a wide array of performance-enhancing policies without risking the system's structural integrity. Examples of such independent policy optimizations include:

- **Intelligent Gossiping:** Designing propagation protocols that prioritize certain messages or prune redundant transmissions to speed up convergence.
- **Graph Sharding:** Partitioning the DCS graph across nodes to handle massive scale.
- **Fast Paths:** Implementing specialized, low-latency protocols for specific, non-contentious interaction patterns.

These optimization avenues, safely insulated from the core correctness logic, represent a rich and promising area for future engineering research.

C. The Scope of Correctness: A Two-Layer Model

It is crucial to precisely define the scope of "correctness" that our theory guarantees to decouple from policy. We

propose a two-layer model for understanding correctness in complex distributed systems:

- **Layer 1: Causal History Correctness.** This foundational layer concerns the integrity of the system's historical record. It asks: "Did all participants agree on a single, unique, and immutable history of what events occurred and their causal relationships?" A system that fails at this layer is fundamentally non-auditable and non-deterministic.
- **Layer 2: Application Semantic Correctness.** This upper layer is built upon a consistent historical record and concerns the business logic. It asks: "Does the sequence of events in the agreed-upon history violate any application-specific invariants (e.g., 'a resource was not double-spent,' 'an account balance never dropped below zero')?"

The core contribution of the DCS theory is to provide a formal guarantee for **Layer 1 Correctness** and to decouple it entirely from operational policies. The DCS serves as the immutable ground truth—the "what happened"—upon which Layer 2 correctness can be verified.

Our theory does not, and cannot, automatically guarantee Layer 2 correctness, as that is the responsibility of the application logic itself. For instance, if an application policy erroneously generates two "spend" contributions for the same resource, the DCS will faithfully and deterministically record both events in the causal history. The history itself will be correct (Layer 1), but it will record a violation of application semantics (Layer 2).

By providing a solid, policy-agnostic foundation for Layer 1, the DCS dramatically simplifies the problem of reasoning about and enforcing Layer 2 correctness, as developers are freed from the complexities of asynchronous message passing and can focus solely on the logical validity of the event sequence itself.

D. The Two Decouplings: Execution Path vs. Recording Mechanism

To fully appreciate the scope of our theory, it is helpful to distinguish between two levels of system behavior, each with its own relationship to "policy":

- **The Execution Path:** This refers to the specific, ordered set of contributions that are actually generated by the agents over time. This path is inherently **policy-dependent**, as it is the direct result of the agents' high-level decision-making or "generation" policies. Different strategies will inevitably lead to different execution paths.
- **The Recording Mechanism:** This refers to the infrastructural process by which an execution path is observed, propagated, and solidified into a global, consistent causal history (the DCS). Our theory proves that this mechanism is **policy-agnostic** with respect to operational policies (routing, scheduling, batching).

The core decoupling achieved by our work is at the level of the Recording Mechanism. We do not eliminate the dependency of the system's outcome on agent strategy; rather, we provide a foundational guarantee that for any strategic path taken by the agents, the recording of that path is deterministic, reliable, and free from the influence of the underlying network

and processing mechanics. This allows for a clean separation of concerns, enabling developers to reason about high-level agent strategy (the Execution Path) on top of a verifiably sound historical foundation (the Recording Mechanism).

E. Limitations and Future Work

While the DCS theory provides a powerful deterministic foundation for distributed intelligent systems, its current theoretical boundaries and future development trajectory must be clearly articulated.

a) *Scope and Future Directions*: The scope of this work is focused on non-Byzantine environments, which is a model that accurately reflects the challenges in a vast domain of collaborative, code-driven MAS. This focus allows us to establish a foundational theory of structural determinism, separate from the orthogonal complexities of adversarial, protocol-violating behavior. Extending these deterministic guarantees to Byzantine fault-tolerant (BFT) settings is therefore a natural and high-priority direction for future research, which would adapt the DCS paradigm for open and permissionless systems.

b) *Future Work: A Clear Roadmap*: Our work opens up several exciting directions for future research:

- (i) **Byzantine-aware DCS Theory**: Extending the DCS theory to Byzantine fault-tolerant (BFT) settings is the highest priority on our roadmap. A promising technical path is to integrate cryptographic primitives with our axiom system. For instance, by requiring each contribution to be digitally signed by its creator, we can strengthen Axioms 3 and 5 against malicious tampering.
- (ii) **Performance Evaluation of a DCS Prototype**: The second major direction is engineering implementation and performance evaluation. We plan to design and implement a prototype DCS system. Based on this prototype, we will conduct quantitative experiments on the various performance optimization strategies discussed previously (e.g., intelligent gossip, sharding, fast paths). This will provide critical performance data and engineering insights for the practical deployment of DCS.
- (iii) **An Application Framework atop DCS**: Ultimately, our goal is to empower application developers with the DCS theory. To this end, we plan to develop a programming framework or library built on top of the DCS theory. This framework will encapsulate the implementation details of the axioms and provide developers with a clean API, enabling them to easily build a new generation of distributed AI applications with verifiability built-in.

VII. CONCLUSION

This paper introduced the *Deterministic Causal Structure* (DCS) as a theoretical foundation for decoupling correctness from policy in multi-agent systems. By axiomatizing contributions, we proved (i) **Existence and Uniqueness** (Theorem 1), (ii) **Policy-Agnostic Invariance** (Theorem 2), (iii) **Observational Equivalence** (Proposition 1), and (iv) **Axiom Minimality** (Theorem 3).

Together these results identify DCS as a *boundary principle* in distributed systems, comparable to CAP or FLP: any system

that aspires to coordination-free determinism must constrain its state model to the expressive power of a join-semilattice. Correctness is thus reframed not as a protocol property, but as a structural invariant of asynchronous computation.

To aid intuition, we included a few *illustrative constructions* that visualize the implications of our theorems. These are not experiments but minimal demonstrations; all guarantees remain purely theorem-driven.

Finally, this perspective suggests the broader design paradigm of **Correctness-as-a-Chassis**: correctness as a fixed, policy-agnostic substrate on which strategies can evolve independently. This principle opens the path toward modular and verifiably safe distributed intelligent systems, establishing DCS as a foundational building block for the mathematics of distributed intelligence.

APPENDIX A

FORMAL PROOFS OF FOUNDATIONAL LEMMAS

Lemma 1 (State Monotonicity). *For any agent i and key k , its local state sequence $\{M_i(k, t)\}_{t \in \mathbb{N}}$ is monotone non-decreasing with respect to the partial order $\sqsubseteq$ of the dcpo-join-semilattice $(L_k, \sqsubseteq, \sqcup)$.*

Proof: The local state update rule is defined as $M_i(k, t+1) \leftarrow M_i(k, t) \sqcup \text{payload}(\delta)$. According to Axiom 2, the state space is a join-semilattice where the join operation $\sqcup$ is inflationary, meaning $x \sqsubseteq x \sqcup y$ for all elements x, y . Therefore, $M_i(k, t) \sqsubseteq M_i(k, t+1)$ holds for every update, establishing the monotonicity of the state sequence. ■

Lemma 2 (Order and Duplicate Irrelevance on Directed Sets). *For any finite or directed set of contributions for a given key k , the final merged state is independent of both the arrival order and the multiplicity of each contribution.*

Proof: Let $\{\delta_j\}_{j \in J}$ be a finite or directed set of contributions. The final state is the join of all payloads: $\sqcup_{j \in J} \text{payload}(\delta_j)$. By Axiom 2, the join operation $\sqcup$ is associative and commutative, which guarantees order-invariance. The idempotence of $\sqcup$ ($x \sqcup x = x$) ensures that duplicate contributions do not alter the result. The directed-completeness property guarantees that this supremum exists for any directed set of contributions. ■

Lemma 3 (Decomposability and Traceability). *An agent's local state is always equal to the join of all unique contribution payloads it has observed. Furthermore, the causal ancestry of any contribution is unambiguously traceable through the immutable 'parents' relation.*

Proof: By construction, the local state $M_i(k)$ is formed exclusively by applying the join operation. By induction and Lemma 2, $M_i(k) = \sqcup_{\delta \in S_i(k)} \text{payload}(\delta)$, where $S_i(k)$ is the set of unique contributions received by agent i for key k . The traceability of causal history is a direct consequence of Axioms 3 and 4, which state that the 'rid' and 'parents' set of each contribution are immutable and globally consistent. This fixes the vertices and edges of the provenance graph at the moment of creation. ■

Lemma 4 (Eventual Propagation). *For any key k , every contribution δ created for that key will eventually be delivered at least once to every agent in the relevant set $\text{Rel}(k)$.*

Proof: This is a direct restatement of Axiom 1 (Localized Weak Fairness). The axiom guarantees that if a contribution is persistently available in the network, all interested agents will eventually receive it. By Lemma 2, subsequent duplicate deliveries are harmless. ■

Lemma 5 (Information Preservation). *Once a payload has been incorporated into an agent's local state via the join operation, its information is never lost or overwritten by subsequent updates.*

Proof: This follows directly from the state monotonicity established in Lemma 1. Since any subsequent state M' is computed as $M' = M \sqcup x$, it is guaranteed that $M \sqsubseteq M'$. Thus, the information contained in the prior state M is preserved as a lower bound of all future states. ■

APPENDIX B

PROOFS OF CORE THEOREMS

Proof of Theorem 1: We prove the three claims separately.

a) 1. *Existence:* The vertices of the graph G^* are the set of all contributions $\mathcal{R}$. By Axiom 3, each contribution has a unique identifier ('rid'). The edges are defined by the immutable 'parents' metadata. Axiom 5 (Causal Well-Formedness) mandates that an agent can only list 'rid's of contributions it has already observed in the 'parents' set of a new contribution. This enforces a strict temporal ordering on edge creation, making it impossible to form cycles. Therefore, G^* is a well-defined Directed Acyclic Graph (DAG).

b) 2. *Uniqueness:* The structure of G^* is determined solely by its vertices ('rid's) and edges ('parents' links). Axioms 3 and 4 guarantee that this metadata is immutable upon creation. Agent policies—such as scheduling, batching, or routing—only affect the *timing* and *path* of information propagation, not the content of the immutable metadata itself. Since the graph's definition is independent of any such policy, its structure is unique up to isomorphism for any given set of contributions.

c) 3. *Constructibility:* For any agent $i \in \text{Rel}(k)$, its local state sequence $\{M_i(k, t)\}$ forms a monotone chain (Lemma 1) in a directed-complete partial order (Axiom 2). This guarantees the existence of a limit state (the supremum). By Lemma 4, agent i will eventually receive every contribution. By Lemma 2, the final joined state is independent of arrival order. Thus, the limit state is unique and equal to the join of all payloads in the final, globally consistent mergeable set. ■

Proof of Theorem 2: This theorem is a direct consequence of Theorem 1. The proof of uniqueness in Theorem 1 already established that the graph's structure is determined by immutable metadata, which is independent of agent policy. Therefore, if the set of contributions is the same, the resulting graphs G_1^* and G_2^* must be isomorphic. The convergence to identical local states follows from the constructibility proof, which showed the limit state is the join over the complete set

of payloads, a calculation that is itself order-independent and thus policy-agnostic. ■

APPENDIX C

PROOFS OF SUPPORTING PROPOSITIONS

Proof of Proposition 1: For computations that only use causal queries (ancestor/descendant) and semilattice-homomorphic aggregations, two executions are observationally indistinguishable if and only if their DCS graphs are isomorphic.

($\Leftarrow$) If the DCS graphs G_1^* and G_2^* are isomorphic, there exists a bijection that preserves `rids`, `parents` relations, and `payloads`. Any causal query on G_1^* will yield a result that maps directly to the identical query on G_2^* . Any aggregation over payloads will operate on identical sets of data and, due to the properties of the join-semilattice, produce identical results. Thus, the executions are observationally indistinguishable.

($\Rightarrow$) If the graphs are non-isomorphic, there must exist a structural difference. For example, a contribution r in G_1^* has a parent p that is not a parent of r in G_2^* . A simple query such as "Is p an ancestor of r ?" would return true in the first execution and false in the second. This constitutes a distinguishable observation. Therefore, observational indistinguishability implies graph isomorphism. ■

Proof of Proposition 2: There exist systems satisfying standard CRDT conditions that converge to the same state value but produce non-isomorphic causal histories.

We prove by construction. Let the state be a set and the merge operation be set union ($\cup$), a valid join-semilattice. Consider two executions generating contributions with payloads $\{x\}$ and $\{y\}$.

Execution 1 (Concurrent): Agent A creates contribution δ_x (payload $\{x\}$), and Agent B concurrently creates δ_y (payload $\{y\}$). Neither is a parent of the other.

- **Final Value:** All agents eventually receive both and compute the final state $\{x\} \cup \{y\} = \{x, y\}$.
- **Causal Structure:** The DCS graph consists of two disconnected nodes, $\{\delta_x, \delta_y\}$, representing concurrent events.

Execution 2 (Causal): Agent A creates δ_x . Agent B observes δ_x and then creates δ_y , explicitly setting `parents`(δ_y) = `rid`(δ_x).

- **Final Value:** All agents eventually receive both and compute the final state $\{x\} \cup \{y\} = \{x, y\}$.
- **Causal Structure:** The DCS graph is a two-node chain, $\delta_x \rightarrow \delta_y$, representing a causal dependency.

Both executions satisfy CRDT value convergence, arriving at the identical state $\{x, y\}$. However, their DCS graphs are non-isomorphic. This demonstrates that the structural guarantee of a DCS is strictly stronger than the value convergence guarantee of CRDTs. ■

APPENDIX D

PROOF OF THEOREM 3 (AXIOM MINIMALITY)

Proof of Theorem 3: The set of Axioms 1 through 5 is minimal. The removal of any single axiom permits a counterexample where the guarantees of Theorem 1 fail.

Sufficiency was established by the proof of Theorem 1. We prove necessity by constructing a counterexample for the removal of each axiom.

Without Axiom 1 (Weak Fairness) An agent $i \in \text{Rel}(k)$ might never receive a contribution δ that other agents have received. Its local state will converge to a different limit than others, violating the **constructibility** guarantee of a unique, globally consistent state.

Without Axiom 2 (Join-Semilattice) If the merge operation is not associative, commutative, and idempotent (e.g., "last-writer-wins"), the final state becomes dependent on the arbitrary network delivery order. This violates the **uniqueness** and **constructibility** guarantees, as different policies lead to different outcomes.

Without Axiom 3 (RID Uniqueness and Immutability)

If two contributions could share the same 'rid', or if a contribution's 'payload' could be mutated after creation, the set of vertices in the global graph G^* would be ill-defined and non-static. This violates the **existence** of a single, well-defined graph.

Without Axiom 4 (Parent Set Immutability) If the 'parents' metadata were mutable, an agent could alter the causal history after the fact. Two different agents could observe and record different parent sets for the same contribution, leading to non-isomorphic graphs. This violates the **uniqueness** guarantee.

Without Axiom 5 (Causal Well-Formedness) An agent could create a contribution δ_1 that names a not-yet-created contribution δ_2 as its parent, while the creator of δ_2 names δ_1 as its parent. This would create a cycle ($\delta_1 \rightarrow \delta_2 \rightarrow \delta_1$). A graph with cycles is not a DAG, violating the fundamental **existence** guarantee of the DCS.

Since removing any axiom breaks at least one core guarantee, the axiom set is minimal. ■

REFERENCES

- [1] L. Lamport, R. Shostak, and M. Pease, "The byzantine generals problem," *ACM Transactions on Programming Languages and Systems*, vol. 4, no. 3, pp. 382–401, 1982.
- [2] B. M. Oki and B. H. Liskov, "Viewstamped replication: A new primary copy method to support highly-available distributed systems," in *Proceedings of the Seventh Annual ACM Symposium on Principles of Distributed Computing*, 1988, pp. 8–17.
- [3] L. Lamport, "The part-time parliament," *ACM Transactions on Computer Systems (TOCS)*, vol. 16, no. 2, pp. 133–169, 1998.
- [4] D. Ongaro and J. Ousterhout, "In search of an understandable consensus algorithm," in *2014 USENIX Annual Technical Conference (USENIX ATC 14)*, 2014, pp. 305–319.
- [5] P. T. Wojciechowski, T. Kobus, and M. Kokociński, "State-machine and deferred-update replication: Analysis and comparison," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 3, pp. 891–904, 2016.
- [6] A. Burgos, E. Alchieri, F. Dotti, and F. Pedone, "Exploiting concurrency in sharded parallel state machine replication," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 9, pp. 2133–2147, 2021.
- [7] T. Kobus, M. Kokociński, and P. T. Wojciechowski, "Hybrid transactional replication: State-machine and deferred-update replication combined," *IEEE Transactions on Parallel and Distributed Systems*, vol. 29, no. 7, pp. 1499–1514, 2018.
- [8] A. Nogueira, A. Casimiro, and A. Bessani, "Elastic state machine replication," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 9, pp. 2486–2499, 2017.
- [9] M. Shapiro, N. Preguiça, C. Baquero, and M. Zawirski, "Conflict-free replicated data types," in *Symposium on Self-Stabilizing Systems*. Springer, 2011, pp. 386–400.
- [10] M. Shapiro, "A comprehensive study of convergent and commutative replicated data types," Inria, Tech. Rep., 2011.
- [11] C. A. Ellis and S. J. Gibbs, "Concurrency control in groupware systems," in *Proceedings of the 1989 ACM SIGMOD international conference on Management of data*, 1989, pp. 399–407.
- [12] C. Gadea, B. Ionescu, and D. Ionescu, "A control loop-based algorithm for operational transformation," in *2020 IEEE 14th International Symposium on Applied Computational Intelligence and Informatics (SACI)*. IEEE, 2020, pp. 000 247–000 254.
- [13] P. Mahajan, L. Alvisi, M. Dahlin *et al.*, "Consistency, availability, and convergence," *University of Texas at Austin Tech Report*, vol. 11, p. 158, 2011.
- [14] P. S. Almeida, A. Shoker, and C. Baquero, "Delta state replicated data types," *Journal of Parallel and Distributed Computing*, vol. 111, pp. 162–173, 2018.
- [15] H. Zarzour and M. Sellami, "B-set: a synchronization method for distributed semantic stores," in *2012 IEEE International Conference on Complex Systems (ICCS)*. IEEE, 2012, pp. 1–6.
- [16] M. Kokociński, T. Kobus, and P. T. Wojciechowski, "On mixing eventual and strong consistency: acute cloud types," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 6, pp. 1338–1356, 2021.
- [17] K. Karlsson, W. Jiang, S. Wicker, D. Adams, E. Ma, R. van Renesse, and H. Weatherspoon, "Vegvisir: A partition-tolerant blockchain for the internet-of-things," in *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2018, pp. 1150–1158.
- [18] A. Auvolet and F. Taïani, "Merkle search trees: Efficient state-based crdts in open networks," in *2019 38th Symposium on Reliable Distributed Systems (SRDS)*. IEEE, 2019, pp. 221–22109.
- [19] L. Lamport, "Time, clocks, and the ordering of events in a distributed system," *Communications of the ACM*, vol. 21, no. 7, pp. 558–565, 1978.
- [20] C. J. Fidge, "Timestamps in message-passing systems that preserve the partial ordering," in *Proceedings of the 11th Australian Computer Science Conference*, 1988, pp. 56–66.
- [21] F. Mattern, "Virtual time and global states of distributed systems," in *Parallel and Distributed Algorithms*, M. Cosnard, Y. Robert, M. Quinton, and P. Raynal, Eds. North-Holland, 1988, pp. 215–226.
- [22] R. Al-Ekram and R. Holt, "Multi-consistency data replication," in *2010 IEEE 16th International Conference on Parallel and Distributed Systems*. IEEE, 2010, pp. 568–577.
- [23] G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall, and W. Vogels, "Amazon's highly available key-value store," vol. 41, no. 6. ACM New York, NY, USA, 2007, pp. 205–220.
- [24] L. Baird, "The swirls hashgraph consensus algorithm: Fair, fast, byzantine fault tolerance," *Swirls Tech Reports SWIRLDS-TR-2016-01, Tech. Rep.*, vol. 34, pp. 9–11, 2016.
- [25] A. Gagol and M. Świątek, "Aleph: A leaderless, asynchronous, byzantine fault tolerant consensus protocol," *arXiv preprint arXiv:1810.05256*, 2018.
- [26] G. Danezis, L. Kokoris-Kogias, A. Sonnino, and A. Spiegelman, "Narwhal and tusk: a dag-based mempool and efficient bft consensus," in *Proceedings of the Seventeenth European Conference on Computer Systems*, 2022, pp. 34–50.
- [27] A. Spiegelman, N. Giridharan, A. Sonnino, and L. Kokoris-Kogias, "Bullshark: Dag bft protocols made practical," in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, 2022, pp. 2705–2718.
- [28] Y. Jo and C. Park, "Enhancing ethereum poa clique network with dag-based bft consensus," in *2024 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. IEEE, 2024, pp. 655–659.
- [29] X. Dai, G. Wang, J. Xiao, Z. Guo, R. Hao, X. Xie, and H. Jin, "Lightdag: A low-latency dag-based bft consensus through lightweight broadcast," in *2024 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. IEEE, 2024, pp. 998–1008.
- [30] Y. Zhou, J. Xiao, X. Dai, and H. Jin, "Plaindag: A low-latency asynchronous dag bft protocol with best-effort broadcast," *IEEE Transactions on Information Forensics and Security*, 2025.
- [31] R. Ladelsky and R. Friedman, "On quorum sizes in dag-based bft protocols," 2025.
- [32] N. Diallo, L. Xu, D. Alsagheer, Y. Lu, and L. Shi, "Optimized consensus with dagwise: A gnn-enhanced approach for scalable and fault-tolerant dag-based bft," in *2025 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. IEEE, 2025, pp. 1–5.

- [33] J. Cheney, L. Chiticariu, W.-C. Tan *et al.*, “Provenance in databases: Why, how, and where,” *Foundations and Trends® in Databases*, vol. 1, no. 4, pp. 379–474, 2009.
- [34] A. Nepal, M. A. Amanullah, R. Doss, and F. Jiang, “Secure data provenance in internet of vehicles with data plausibility for security and trust,” in *2024 IEEE World AI IoT Congress (AIoT)*. IEEE, 2024, pp. 612–618.
- [35] A. Nepal, R. Doss, and F. Jiang, “Secure data provenance in internet of vehicles with verifiable credentials for security and privacy,” in *2024 54th Annual IEEE/IFIP International Conference on Dependable Systems and Networks-Supplemental Volume (DSN-S)*. IEEE, 2024, pp. 59–61.
- [36] M. Miller, S. Williams, G. G. Dagher, and M. Long, “Prism: A blockchain-enabled reputation-based consensus for enhancing scientific workflow provenance,” in *2023 IEEE 9th International Conference on Collaboration and Internet Computing (CIC)*. IEEE, 2023, pp. 72–81.
- [37] S. Nakamoto, “Bitcoin: A peer-to-peer electronic cash system,” 2008.
- [38] V. Buterin *et al.*, “A next-generation smart contract and decentralized application platform,” pp. 2–1, 2014.
- [39] S. King and S. Nadal, “Ppcoin: Peer-to-peer crypto-currency with proof-of-stake,” vol. 19, no. 1, 2012.
- [40] T. Wang, Q. Wang, Z. Shen, Z. Jia, and Z. Shao, “Understanding intrinsic characteristics and system implications of dag-based blockchain,” in *2020 IEEE International Conference on Embedded Software and Systems (ICESS)*. IEEE, 2020, pp. 1–6.
- [41] S. Yang, Z. Chen, L. Cui, M. Xu, Z. Ming, and K. Xu, “Codag: An efficient and compacted dag-based blockchain protocol,” in *2019 IEEE International Conference on Blockchain (Blockchain)*. IEEE, 2019, pp. 314–318.
- [42] S. Zhang, J. Xiao, E. Wu, F. Cheng, B. Li, W. Wang, and H. Jin, “Morphdag: A workload-aware elastic dag-based blockchain,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 36, no. 10, pp. 5249–5264, 2024.
- [43] G. Liao, H. Ding, C. Zhong, and Y. Lei, *IEEE Internet of Things Journal*, vol. 11, no. 20, pp. 32 759–32 772, 2024.