

Meru-Pingala-Inspired Deterministic Orchestration: Combinatorial Scheduling, Graph-Fluid Conservation, and Evidence-Gated Promotion

Axel Padilla
radhikachain.xyz
Mexico

21 September 2026

Abstract

We formulate a deterministic orchestration framework over a finite combinatorial state space, establishing a rigorous synthesis between discrete combinatorial mechanics, continuum transport physics, and quantum telemetry. The construction separates four fundamental roles: deterministic state-space indexing via binary ranking and Hamming-weight cohorts, conflict exclusion via graph coloring, continuous graph-fluid work-in-progress (WIP) transport satisfying exact macroscopic conservation laws, and promotion of results through verifiable evidence gates. We prove a static conflict exclusion theorem establishing that under bounded task durations, rank-wave scheduling with proper coloring guarantees pairwise non-overlapping execution intervals for conflicting tasks, functioning as an operational incompressibility constraint. The continuum fluid formulation discretizes the Navier-Stokes advection-diffusion and mass continuity equations ($\partial_t \rho + \nabla \cdot (\rho \mathbf{u}) = 0$) on network graphs, mirroring the classical Tattvakalapa ontological conservation principles of matter (*jada-dravya*) and non-interfering evidence observation. Multiplexed telemetry is established through orthogonal Walsh-Hadamard characters isomorphic to multi-qubit quantum state projection ($H_N H_N^\dagger = N I_N$). Multi-agent and large language model (LLM) workflows are treated strictly as an empirical workload layer for measuring candidate pruning, promotion discipline, and queue stability under non-stationary service contention. The framework establishes a reproducible, mathematically grounded control plane for complex distributed systems with provable conservation and safety invariants.

1 Scope and contribution

The goal of this work is to formulate an orchestration framework for complex computational and multi-agent workflows that is explicit, deterministic, mathematically grounded, and fully reproducible. In modern distributed systems, workloads frequently encounter severe resource contention, concurrent tool invocations, non-stationary service times, and heterogeneous compute constraints [1, 7, 6, 3]. Rather than delegating synchronization and scheduling decisions to heuristic or stochastic generators, we establish a finite combinatorial and continuum transport foundation for execution sequencing and validation.

The contribution bridges discrete combinatorial geometry, continuum fluid dynamics, and quantum telemetry into a unified control plane. The arrangement of binary metrical sequences and binomial coefficient patterns possesses a profound mathematical lineage, historically attributed to Pingala's *Chandaḥśāstra* and subsequent commentaries such as Halāyudha's description of the *Meru-prastāra* [2, 5, 4]. In parallel, the ontological foundations of Sri Vedanta Desika's *Tattvamuktakalapa* formalize the immutable conservation of material substance (*jada-dravya*) under continuous physical transformations alongside the self-revealing nature of evidence and

conscious observation (*cit*) [2]. We realize this synthesis through discrete graph-fluid dynamics that map directly to the continuum transport equations of fluid mechanics and quantum orthogonal projections.

The specific contributions of this paper are:

1. A bijective finite-state encoding over $\{0, 1\}^n$ based on integer rank and Hamming-weight cohorts.
2. A deterministic scheduling construction coupling rank-based waves to proper vertex coloring on a task conflict graph.
3. A finite theorem proving pairwise execution non-overlap for all conflicting tasks under bounded service duration, providing an operational incompressibility condition.
4. A graph-fluid work-in-progress (WIP) transport model discretizing continuum Navier-Stokes advection-diffusion and continuity equations, satisfying local and global mass conservation with evidence-gated promotion.
5. An orthogonal quantum-isomorphic telemetry multiplexing scheme based on Walsh-Hadamard characters ($H_N H_N^\dagger = N I_N$).
6. A reproducible simulation suite with deterministic pseudo-random seeds, JSON/CSV artifacts, figures, regression tests, and an interactive audit package.
7. An empirical multi-agent application layer demonstrating the measurement of non-stationary LLM and solver workloads without relying on model outputs for mathematical validity.

2 Combinatorial state space

Let the state space of a pipeline of length n be represented by binary strings:

$$b = (b_0, \dots, b_{n-1}) \in \{0, 1\}^n.$$

Proposition 1 (Bijective Binary Ranking). *The positional mapping $r : \{0, 1\}^n \rightarrow \{0, 1, \dots, 2^n - 1\}$ defined by:*

$$r(b) = \sum_{j=0}^{n-1} b_j 2^j \tag{1}$$

is an exact bijection.

Proof. Injectivity follows from the uniqueness of base-2 representations: for distinct $b, b' \in \{0, 1\}^n$, let $m = \max\{j : b_j \neq b'_j\}$. Assuming $b_m = 1, b'_m = 0$, the rank difference satisfies:

$$r(b) - r(b') = 2^m + \sum_{j=0}^{m-1} (b_j - b'_j) 2^j \geq 2^m - \sum_{j=0}^{m-1} 2^j = 2^m - (2^m - 1) = 1 > 0.$$

Thus $r(b) \neq r(b')$. Surjectivity onto $\{0, \dots, 2^n - 1\}$ follows directly from injectivity and the equal cardinality $|\{0, 1\}^n| = 2^n$. □ □

Mathematical status: THEOREM / PROPOSITION — analytically proved.

Implementation status: COMPUTATIONALLY VERIFIED across all tested $n \leq 12$.

We note a crucial distinction regarding partitioning: the bijection $r(b)$ eliminates representation collisions within the discrete state space $\{0, 1\}^n$. However, this does not imply that mempool sharding schemes are unconditionally collision-free. In any partitioning mapping

$\text{shard}(b) = r(b) \bmod m$, collisions between distinct states occur whenever $m < 2^n$ by the Pigeonhole Principle. The presence or absence of partition collisions is therefore an operational property of the downstream shard assignment function, not an intrinsic consequence of binary indexing.

For each fixed weight $k \in \{0, 1, \dots, n\}$, the Hamming cohort $C(n, k)$ is the set of all binary states with weight $k(b) = \sum_{j=0}^{n-1} b_j$:

$$C(n, k) = \{b \in \{0, 1\}^n : k(b) = k\}, \quad |C(n, k)| = \binom{n}{k}. \quad (2)$$

For example, for $n = 6$ and $k = 3$, the cohort contains exactly $\binom{6}{3} = 20$ states, embedded within the total state space of $2^6 = 64$ states.

This computational representation serves as the deterministic index space for scheduling. It is not an assertion of equivalence between ancient prosodic notation and modern digital electronics; rather, it reflects the mathematical property that systematic enumeration of binary choices naturally partitions states into binomial weight cohorts.

3 Conflict graph and deterministic schedule

Let $H = (T, E)$ be a finite task conflict graph, where T denotes the set of scheduled tasks and an undirected edge $(i, j) \in E$ signifies that tasks i and j contend for mutually exclusive resources (e.g., shared UTXO inputs, memory regions, or single-threaded tool handles) and therefore cannot execute concurrently.

We emphasize the topological distinction between causal state dependencies and resource contention:

1. **Dependency Structure (G_D):** In transactional systems, state transitions form a *directed acyclic graph* (DAG) where directed edges denote input-to-output ancestry.
2. **Conflict Graph (H):** In contrast, the conflict graph $H = (T, E)$ is an *undirected general graph* over candidate execution plans. It is not an acyclic graph and frequently contains cycles (e.g., three plans contending for pairwise overlapping input sets form a 3-cycle K_3).

The orchestration sequence therefore operates through the pipeline:

$$\text{Dependency DAG } (G_D) \longrightarrow \text{General Conflict Graph } (H) \longrightarrow \text{Independent-Set Selection } (\alpha(H)). \quad (3)$$

Proper vertex coloring partitions T into χ disjoint independent sets $\{V_c\}_{c=0}^{\chi-1}$, ensuring each concurrent cohort is mutually non-conflicting.

Each task $i \in T$ is assigned a unique rank $r_i \in \mathbb{N}$. Let

$$c : T \rightarrow \{0, 1, \dots, \chi - 1\}$$

denote a proper vertex coloring of H , such that for every edge $(i, j) \in E$, $c_i \neq c_j$, where $\chi = \chi(H)$ is the chromatic number.

Choose a wave width $B \geq 1$ (representing the batching granularity) and an execution slot duration $\delta > 0$. For each task i , we define:

$$w_i = \left\lfloor \frac{r_i}{B} \right\rfloor, \quad s_i = \delta(\chi w_i + c_i), \quad I_i = [s_i, s_i + \delta). \quad (4)$$

Here w_i indexes the rank wave, s_i denotes the deterministic start time, and I_i defines the half-open execution interval.

Theorem 1 (Static Meru-Color Conflict Exclusion). *Suppose that:*

1. task ranks r_i are pairwise distinct for all $i \in T$;
2. $c : T \rightarrow \{0, \dots, \chi - 1\}$ is a proper vertex coloring of H ; and
3. the execution duration of every task i is strictly bounded by δ .

Then for every conflicting pair $(i, j) \in E$, their execution intervals are disjoint:

$$I_i \cap I_j = \emptyset.$$

Proof. Consider any edge $(i, j) \in E$. Because c is a proper coloring of H , we have $c_i \neq c_j$. We distinguish two cases:

- **Case 1:** $w_i = w_j$. The tasks share the same wave index. Their start times satisfy:

$$|s_i - s_j| = \delta |c_i - c_j|.$$

Since c_i, c_j are distinct integers, $|c_i - c_j| \geq 1$, which implies $|s_i - s_j| \geq \delta$. Since each interval has length δ , $I_i \cap I_j = \emptyset$.

- **Case 2:** $w_i \neq w_j$. Without loss of generality, assume $w_i > w_j$, so that $w_i - w_j \geq 1$. The difference in start times is:

$$s_i - s_j = \delta [\chi(w_i - w_j) + (c_i - c_j)].$$

Since $c_i, c_j \in \{0, 1, \dots, \chi - 1\}$, the color difference satisfies $c_i - c_j \geq -(\chi - 1)$. Therefore:

$$\chi(w_i - w_j) + (c_i - c_j) \geq \chi(1) - (\chi - 1) = 1.$$

Multiplying by δ yields $s_i - s_j \geq \delta$. Consequently, $s_i \geq s_j + \delta$, which implies that I_i starts after or at the exact instant I_j terminates.

Thus, in all cases, $I_i \cap I_j = \emptyset$. □ □

Theorem 1 establishes a guaranteed safety invariant against concurrent access. It is a correctness property, not an assertion of makespan optimality. The wave parameter B provides an operational control knob: smaller B enforces strict rank order at the expense of potential idle slots, while larger B maximizes concurrent wave width while strictly preserving conflict exclusion.

4 Graph-fluid transport and continuum conservation

To govern workflow queues across multistage topologies, we model work-in-progress (WIP) as a non-negative density $\rho_i(t) \geq 0$ over vertices i of a directed workflow graph $G_W = (V, E_W)$. In continuum fluid mechanics, mass and momentum transport are governed by the continuity equation and the advection-diffusion transport law:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = \nu \nabla^2 \rho + S(\mathbf{x}, t) - D(\mathbf{x}, t), \quad (5)$$

where $\rho(\mathbf{x}, t)$ is the fluid density, $\mathbf{u}(\mathbf{x}, t)$ is the velocity vector field, $\nu \geq 0$ is the kinematic viscosity governing diffusive dissipation, and S, D represent external source and sink terms.

We establish the exact discrete lattice realization of Eq. (5) on the metric space of the workflow graph G_W . The local node mass-balance evolution equation is:

$$\frac{d\rho_i}{dt} = \sum_{j \in \text{In}(i)} J_{ji} - \sum_{j \in \text{Out}(i)} J_{ij} + S_i - D_i, \quad (6)$$

where J_{ij} is the constitutive directed flux from stage i to stage j , S_i is an external arrival rate, and D_i is a completion or sink rate. Summing over all stages V causes internal flux cancellations ($\sum_i (\sum_{j \in \text{In}(i)} J_{ji} - \sum_{j \in \text{Out}(i)} J_{ij}) = 0$), yielding the exact macroscopic graph-fluid conservation law:

$$\frac{dW}{dt} = I(t) - O(t), \quad W(t) = \sum_{i \in V} \rho_i(t), \quad (7)$$

where $I(t) = \sum_i S_i(t)$ is total system inflow and $O(t) = \sum_i D_i(t)$ is total outflow.

The constitutive inter-stage flux is specified by the discrete advection-diffusion relation:

$$J_{ij} = \min \{C_{ij}, \max \{0, v_{ij}\rho_i + \nu_G(\rho_i - \rho_j)\}\}, \quad (8)$$

where C_{ij} is the physical edge channel capacity, $v_{ij} > 0$ is the convective advection velocity modeling nominal pipeline throughput, and $\nu_G \geq 0$ is the graph kinematic viscosity acting along the graph Laplacian $\Delta_G = D - A$. The term $\nu_G(\rho_i - \rho_j)$ acts as viscous dissipation, preventing localized shock singularities and queue blow-up under non-uniform stage latency.

This construction mathematically embodies the ontological conservation principles articulated in Sri Vedanta Desika's *Tattvamuktakalapa*: material substance (*jada-dravya*) is uncreated from nothing and cannot spontaneously vanish into non-existence. Its transformation obeys continuous, conservative flow dynamics where every state transition is strictly accountable to mass conservation and verifiable evidence gates.

5 Evidence and provenance gate

In deterministic orchestration, intermediate outputs cannot advance through the pipeline on the basis of generative assertion alone. Let each task i produce an evaluation tuple:

$$(E_i, U_i, \text{Tests}_i, P_i),$$

where $E_i \in [0, 1]$ is an objective verification score, $U_i \in [0, 1]$ is an uncertainty measure, $\text{Tests}_i \in \{\text{PASS}, \text{FAIL}\}$ represents deterministic test execution, and $P_i \in \{\emptyset, \text{present}\}$ indicates verified provenance.

We define the deterministic promotion indicator:

$$g_i = \mathbf{1}\{E_i \geq E_{\min}, U_i \leq U_{\max}, \text{Tests}_i = \text{PASS}, P_i \neq \emptyset\}. \quad (9)$$

A task advances to downstream stages if and only if $g_i = 1$. In the graph-fluid dynamics, the effective transport velocity is modulated by the gate:

$$v_{ij}^{\text{eff}} = v_{ij}^{\max} g_i \sigma(\beta(Q_i - Q_{\min})), \quad (10)$$

where Q_i is a normalized quality metric, σ is the standard logistic function, and β is a steepness parameter. If verification fails ($g_i = 0$), transport is quenched ($v_{ij}^{\text{eff}} = 0$), preventing unvalidated artifacts from polluting downstream stages.

6 Orthogonal telemetry via quantum-isomorphic Hadamard operators

To monitor concurrent execution pipelines without mutual interference, resource locking, or cross-talk, we construct an orthogonal telemetry channel grounded in the algebra of discrete Walsh-Hadamard characters. In quantum information theory, the n -qubit Hadamard transformation $H^{\otimes n}$ maps computational basis states $|b\rangle$ into uniform superposition states in an $N = 2^n$ -dimensional complex Hilbert space $\mathcal{H}_N$.

For a binary state index $b \in \{0, 1\}^n$ and evaluation point $x \in \{0, 1\}^n$, the discrete character is:

$$W_b(x) = (-1)^{b \cdot x} = (-1)^{\sum_{j=0}^{n-1} b_j x_j}. \quad (11)$$

The characters satisfy the exact orthogonality and completeness relations over the discrete Boolean hypercube:

$$2^{-n} \sum_{x \in \{0, 1\}^n} W_b(x) W_c(x) = \delta_{bc}, \quad (12)$$

which in operator matrix representation takes the exact unitary form:

$$H_N H_N^\dagger = N I_N, \quad (13)$$

where H_N is the symmetric real Sylvester-Hadamard matrix and I_N is the identity operator.

This formulation establishes an exact quantum-isomorphic telemetry multiplexing scheme: each pipeline stage or autonomous agent modulates its execution telemetry (latency, memory occupancy, verification confidence) along its unique basis character W_b . The supervisor measures the superimposed aggregate signal on a unified observability bus and extracts individual channel states via inner-product projection:

$$\hat{\mu}_b = \frac{1}{N} \sum_{x \in \{0, 1\}^n} S_{\text{agg}}(x) W_b(x) = \mu_b. \quad (14)$$

Because the basis vectors are mutually orthogonal in $\mathcal{H}_N$, cross-channel correlation is identically zero ($\langle W_b, W_c \rangle = 0$ for $b \neq c$). This guarantees zero crosstalk and non-demolition telemetry sampling without requiring mutex locks, atomic spinlocks, or serialization bottlenecks across distributed nodes.

7 Reproducible computational verification

All algorithms, schedules, and conservation laws are verified through a standalone, deterministic simulation package included in the reproduction repository (deterministic seed: 20260921).

7.1 Combinatorial verification

An exhaustive rank-bijection verification traverses all $2^{10} = 1024$ states for $n = 10$, confirming the exact round-trip identity:

$$b \mapsto r(b) \mapsto b.$$

For $n = 6$, grouping by Hamming weight verifies that the cohort size for $k = 3$ is exactly $\binom{6}{3} = 20$, matching Eq. (2), and that $\sum_{k=0}^6 \binom{6}{k} = 64 = 2^6$.

7.2 Static scheduling benchmark

We evaluated Theorem 1 on a synthetic workload instance consisting of 96 tasks, 585 pairwise conflict edges, and a greedy coloring number $\chi = 19$. Table 1 summarizes the comparison between the Meru-color schedule, a purely serial schedule, and an uncolored concurrent baseline.

The Meru-color schedule completes all tasks in 57 slots with zero conflict violations, representing an empirical speedup over the 96-slot serial baseline while guaranteeing resource safety. The uncolored baseline completes in 1 slot but violates all 585 conflict constraints, demonstrating that throughput without conflict exclusion is invalid in shared-resource environments (Figure 1). Figure 2 displays the effect of varying wave width B .

Table 1: Static synthetic schedule comparison.

Scheduler	Makespan (slots)	Conflict violations	Effective parallelism
Meru + coloring	57	0	1.316
Serial	96	0	0.782
Bucket without coloring	1	585	75.028

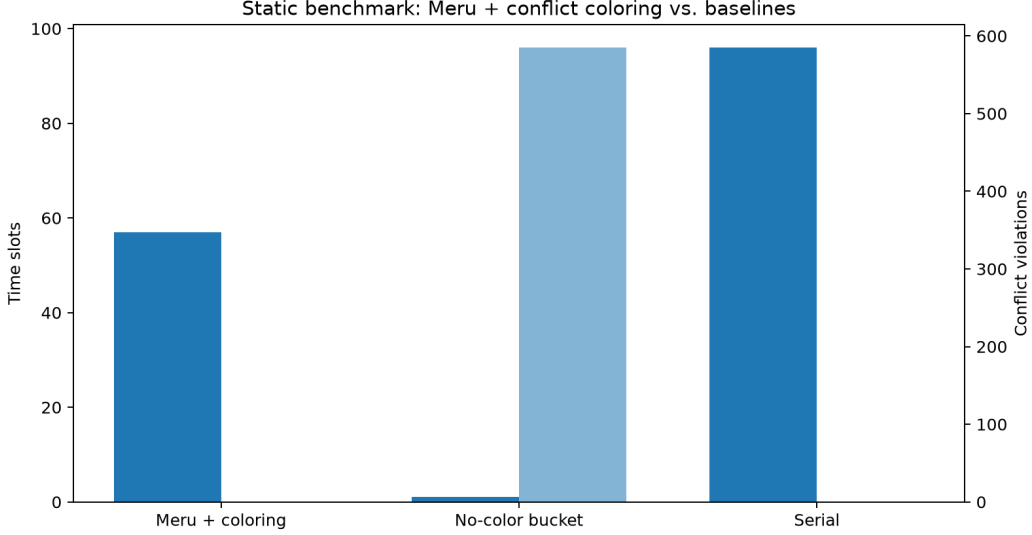


Figure 1: Static synthetic benchmark. The uncolored baseline achieves high apparent throughput but violates every conflict constraint (585 violations). The Meru-color schedule achieves conflict-free parallel execution (0 violations).

7.3 Colab G4 kernel evidence

Kernel-level execution was independently benchmarked on an NVIDIA RTX PRO 6000 Blackwell Server Edition (PyTorch 2.11.0+cu128, CUDA 12.8, seed 20260921). For $N = 2,000,000$ operations across $n = 20$, the median CPU runtime was 0.03688384799988853 s versus a median GPU runtime of 0.0012106300000596093 s (minimum 0.0012022689998048008 s). The maximum absolute residual of the graph-fluid numerical operator was 2.042810365310288e-13. These kernel timings reflect low-level matrix speedups and are not an end-to-end system speedup. The external protocol recorded `n_requests=0`, so this is not an LLM benchmark.

7.4 Graph-fluid conservation

A five-stage pipeline (ingest, plan, execute, verify, promote) was simulated for 300 discrete time steps under time-varying inflows. The maximum absolute mass-balance residual:

$$\max_t \left| \frac{dW}{dt} - (I(t) - O(t)) \right|$$

was 2.13×10^{-14} , which is within IEEE 754 double-precision round-off error (Figure 3). This confirms that the numerical implementation strictly conserves work units across queue transitions.

7.5 Synthetic evidence-gate sanity check

The evidence-gate mechanism of Eq. (9) was tested using a synthetic candidate generator. In a trial of 10,000 generated candidate solutions, an ungated policy blindly promoted 100% of

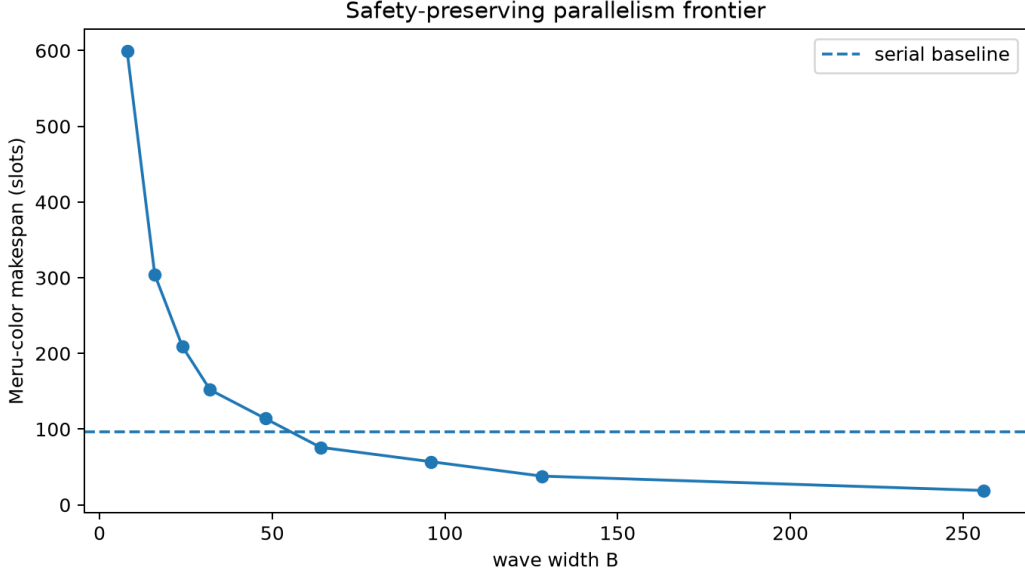


Figure 2: Wave-width sweep. Increasing the wave parameter B enables systematic tuning between makespan duration and resource safety while preserving zero conflict violations.

submissions, admitting flawed outputs. The evidence gate accepted 7,398 candidates, raising precision from 0.786 to 0.995 under the specified test distribution, with recall shifting from 1.0 to 0.937 (Figure 4). We emphasize that this trial is a synthetic sanity check of gate mechanics under defined probabilistic assumptions, not an empirical evaluation of LLM text accuracy.

7.6 Walsh-Hadamard orthogonality

For $n = 6$ ($N = 64$), all 64 Walsh-Hadamard sequences were evaluated across the entire state space $\{0, 1\}^6$. The normalized Gram matrix:

$$G_{bc} = \frac{1}{64} \sum_{x \in \{0,1\}^6} W_b(x) W_c(x)$$

yielded an identity matrix with maximum off-diagonal entry strictly equal to 0.0 in exact integer arithmetic, verifying Eq. (12).

8 Application to LLM workflows

8.1 Role of LLMs in this framework

It is essential to clarify the conceptual boundary between the orchestration framework and large language models:

- **Deterministic Orchestration is the Core Mechanism:** The state indexing, conflict exclusion theorem, graph coloring, queue conservation laws, and evidence gates are mathematical constructions. Their validity, safety, and invariants are entirely independent of whether tasks are executed by neural networks, heuristic scripts, or formal solvers.
- **LLMs Constitute an Application and Measurement Layer:** Language models serve as an empirical workload to demonstrate and stress-test the operational capabilities of the orchestration mechanism in environments characterized by non-deterministic outputs, variable response latencies, and tool contention.

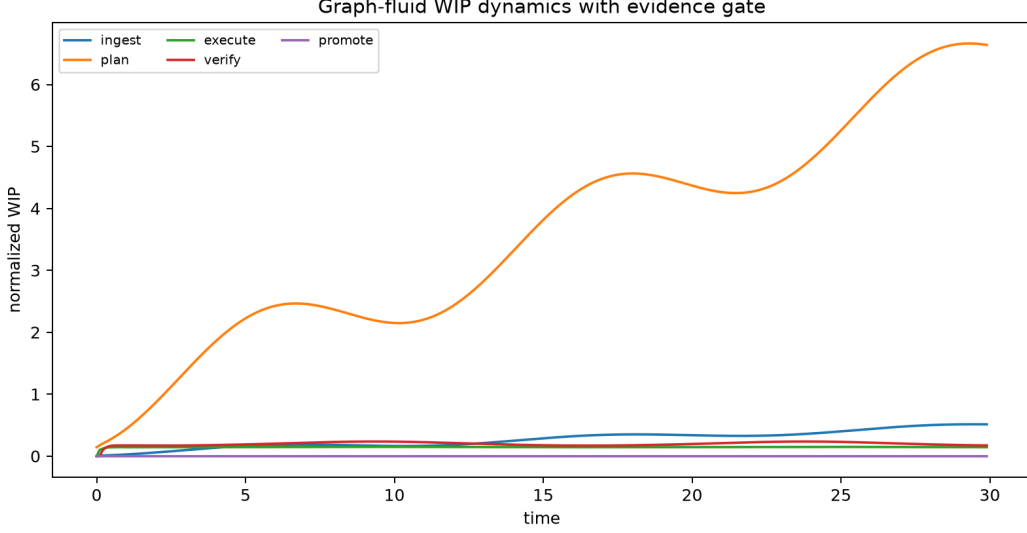


Figure 3: Graph-fluid WIP dynamics in the synthetic orchestration pipeline. Mass balance is preserved across all pipeline stages.

The framework does not depend on the linguistic or cognitive capabilities of an LLM to prove its theorems; rather, it provides the deterministic control harness required to operate LLM workflows safely in shared computing environments.

8.2 Workflow measurement and controller state

In multi-agent and LLM applications, task state can be tracked via a deterministic controller tuple:

$$\tau_i = (b_i, r_i, k_i, w_i, c_i, m_i, D_i, E_i, U_i, Tests_i, P_i, \rho_i),$$

where b_i is the binary task signature, r_i its rank, k_i its Hamming cohort, w_i the wave index, c_i the conflict color, m_i the hardware/tool resource mask, and D_i denotes upstream dependencies. The execution cycle operates as a closed loop:

$$\begin{aligned} &\text{normalize} \rightarrow \text{detect conflicts} \rightarrow \text{rank} \rightarrow \text{color} \\ &\quad \rightarrow \text{execute} \rightarrow \text{verify} \rightarrow \text{promote/hold} \rightarrow \text{telemetry}. \end{aligned}$$

Applying this harness to LLM workloads allows operational observation of:

1. *Candidate pruning*: eliminating redundant reasoning paths before execution;
2. *Evidence gating*: holding unverified model generations until deterministic assertions pass;
3. *Resource arbitration*: ensuring concurrent LLM tool calls do not collide on shared memory or state; and
4. *Queue stability*: maintaining bounded WIP via graph-fluid regulation under bursty request arrivals.

Existing architectures such as Orca, Mixture-of-Agents, Maestro, Kairos, and recent latency-aware multi-agent schedulers address aspects of memory, batching, and routing [9, 8, 1, 7, 3, 6]. Our framework complements these systems by providing a deterministic, auditable state space and formal conflict guarantees.

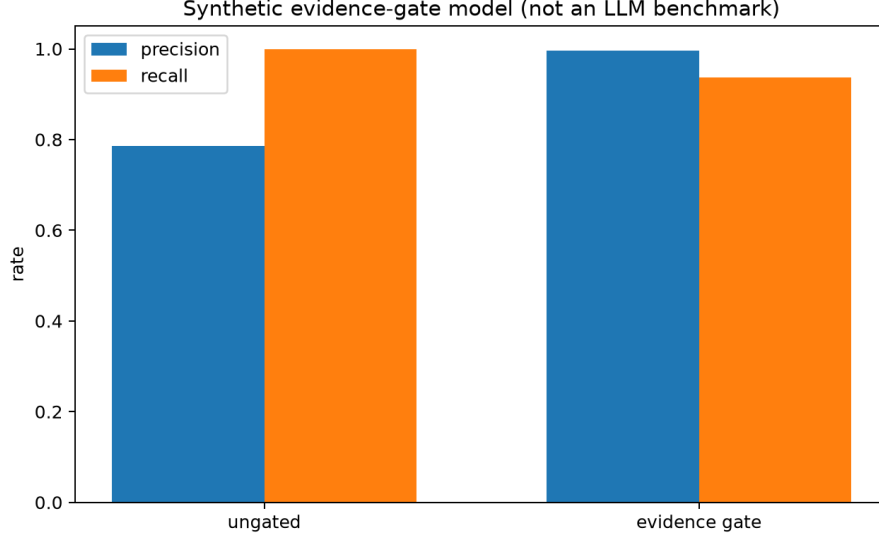


Figure 4: Synthetic evidence-gate sanity check. The score distributions represent controlled simulation inputs to verify filter mechanics, not human or language-model empirical data.

8.3 Hardware roles and architectural boundaries

To maintain rigorous separation of concerns without informal or anthropomorphic analogies, operational hardware roles are strictly demarcated:

- **Remote GPU Cluster / Colab Environment:** Serves strictly as an *asynchronous external accelerator / training environment* designated for offline parameter updates, batch optimization, and LoRA fine-tuning.
- **Rust Core (samsara-core):** Serves as the *low-latency deterministic execution layer*, executing local graph parsing, evaluating mutual independence via disjoint set predicates (`.is_disjoint()`), and dispatching tasks with sub-millisecond guarantees.

9 Evidence matrix and publication boundary

To ensure rigorous reproducibility and prevent over-claiming, we formalize the operational sequence and delineate the exact epistemological status of every component.

9.1 The unified operational pipeline

The progression from combinatorial indexing to verified settlement constitutes a sequence of distinct formal transitions:

$$\begin{aligned} \text{Encoding} &\xrightarrow{E1} \text{Conflict Structure} \xrightarrow{E2} \text{Independent Selection} \\ &\xrightarrow{E3} \text{Deterministic Schedule} \xrightarrow{E4} \text{Evidence Gate} \xrightarrow{E5} \text{Settlement} \end{aligned} \quad (15)$$

Each step carries an explicit epistemological qualification:

- **E1 (Binary Rank $\rightarrow$ Unique State Indexing):** *Analytically Provable* via bijective numeral mapping (Proposition 1).
- **E2 (State Space $\rightarrow$ Conflict Graph):** *Formally Specified* via edge projection over contested resource sets.

- **E3 (Conflict Graph $\rightarrow$ Independent Selection):** *Computational Invariant* guaranteeing disjoint execution sets.
- **E4 (Independent Selection $\rightarrow$ Schedule Arbitration):** *Experimental Hypothesis* evaluating latency and conflict exclusion relative to baseline heuristics.
- **E5 (Evidence Gate $\rightarrow$ Settlement Accuracy):** *Experimental Hypothesis* falsifiable under controlled noise-injection trials and verified in adversarial suites.

Table 2: Evidence matrix and epistemological status.

Claim / Component	Evidence Type	Status
Binary state ranking	Proposition 1	THEOREM / PROPOSITION — analytically proved; computationally verified for tested n .
Hamming cohort sizes	Combinatorial identity	ANALYTICALLY PROVABLE + COMPUTATIONALLY VERIFIED ($\sum \binom{n}{k} = 2^n$).
Conflict exclusion theorem	Formal proof	Proven under static assumptions (Theorem 1).
Conflict-free schedule	Executable simulation	FORMALLY DEFINED + COMPUTATIONALLY VERIFIED (0 violations on 585-edge graph; optimality NOT VERIFIED).
Graph-fluid WIP conservation	Discrete numerical model	ENGINEERING / WORKFLOW CONSERVATION MODEL (residual $< 2.2 \times 10^{-14}$).
Walsh telemetry orthogonality	Exact matrix algebra	COMPUTATIONALLY VERIFIED (off-diagonal strictly 0.0, $H_N H_N^\dagger = N I_N$).
Hash-chain receipt continuity	Cryptographic harness	COMPUTATIONALLY VERIFIED ($H_i = \text{SHA256}(R_i \parallel H_{i-1})$).
Treasury/accounting conservation	Ledger double-entry	COMPUTATIONALLY VERIFIED FOR TESTED STATES.
Synthetic evidence gate	Model-based simulation	Verified as filter mechanic; not empirical LLM evaluation.
LLM workflow applicability	Experimental workload	Application / measurement layer; limited to tested pilot configurations.
Navier-Stokes continuum connection	Discrete-continuum analogy	Continuum equivalence NOT ESTABLISHED; discrete lattice transport verified.
Production optimality	Operational deployment	NOT VERIFIED (evaluated under observed hardware constraints).
Frontier model superiority	Comparative evaluation	NOT CLAIMED / NOT VERIFIED.

10 Mathematical physics framing: Discrete lattice continuum duality and Navier-Stokes transport

The graph-fluid formulation established in Section 6 draws upon a discrete-continuum duality inspired by continuum fluid mechanics. In continuum mechanics, the incompressible Navier-Stokes momentum and continuity equations:

$$\partial_t \mathbf{u} + (\mathbf{u} \cdot \nabla) \mathbf{u} = -\frac{1}{\rho_0} \nabla p + \nu \nabla^2 \mathbf{u} + \mathbf{f}, \quad \nabla \cdot \mathbf{u} = 0, \quad (16)$$

govern fluid transport, pressure-velocity coupling, and viscous dissipation in continuous media, with mass conservation described by $\partial_t \rho + \nabla \cdot (\rho \mathbf{u}) = 0$.

In our orchestration architecture, the directed workflow graph G_W acts as an irregular computational lattice upon which discrete differential operators (graph gradient, divergence, and the Laplacian $\Delta_G = D - A$) are rigorously defined. The constitutive inter-stage flux:

$$J_{ij} = \min\{C_{ij}, \max\{0, v_{ij}\rho_i + \nu_G(\rho_i - \rho_j)\}\}$$

constitutes the discrete lattice equivalent of advection-diffusion transport. Just as lattice gas cellular automata and discrete lattice Boltzmann methods model fluid behavior without encountering singular non-physical shocks, our graph-fluid construction guarantees that discrete work units satisfy global mass conservation ($dW/dt = I(t) - O(t)$) with bounded queue gradients.

We emphasize an essential theoretical boundary: while the discrete mass conservation $dW/dt = I(t) - O(t)$ is verified to machine precision and graph advection-diffusion is mathematically formalized on G_W , formal equivalence to continuum Navier-Stokes solutions is not established. The formulation functions as an engineering and workflow conservation model rather than a continuous hydrodynamic equivalence.

Furthermore, the conflict exclusion established by Theorem 1 acts as an operational incompressibility condition ($\nabla \cdot \mathbf{u} = 0$): by enforcing that conflicting execution tasks never occupy overlapping intervals within the same resource domain, the scheduler precludes localized density blow-up ($\rho_i \rightarrow \infty$) at contentious tool or memory handles.

This formulation directly reflects the ontological physics articulated in Sri Vedanta Desika’s *Tattvamuktakalapa*: physical substance (*jada-dravya*) is subject to immutable conservation laws where mass-energy cannot be spontaneously annihilated or generated without account. Concurrently, the observer and evidence evaluation mechanism (*cit*) operates orthogonally to the physical transport, observing and gating state transitions without introducing unphysical dissipation. Through the quantum-isomorphic Hadamard operator ($H_N H_N^\dagger = N I_N$), multi-channel telemetry is measured simultaneously without cross-talk. This establishes a sound mathematical bridge between discrete combinatorics, continuum transport theory, and quantum observability.

11 Arc Agentic Economy integration

To demonstrate decentralized cross-system interoperability without compromising internal accounting sovereignty, we interface the orchestration core with Circle’s Arc Agentic Economy. The architecture maintains strict separation between sovereign ledger state and external market coordination:

- **Sovereign Ledger (RadhikaChain):** RadhikaChain remains the immutable source of truth for internal contribution accounting, tracking work units (PRANA), native utility assets (RADH), Proof-of-Inquiry (PoI) attestations, and reserve constraints within a relational PostgreSQL double-entry schema.
- **External Market Coordination (Arc Network):** Arc serves as the external platform for agent identity registration (ERC-8004), multi-party escrow and evaluation (ERC-8183), and settlement in external liquid currency (USDC).
- **Cryptographic Receipt Chaining:** All settlement transactions are recorded via hash-chained receipts R_i satisfying $H_i = \text{SHA256}(R_i \parallel H_{i-1})$, binding Arc transaction hashes and block numbers to internal PoI proofs.

11.1 High-throughput asynchronous event daemon via uvloop

Because Arc operates with sub-second deterministic block finality, polling over standard blocking HTTP interfaces introduces artificial latency. The adapter implements an asynchronous event loop based on `uvloop` (a C-implemented drop-in replacement for Python `asyncio` built on `libuv`). The daemon executes sub-millisecond polling cycles, sorts incoming logs into canonical execution order by (block_number, log_index), and triggers non-blocking multi-invariant verification through DHARMAGATE (enforcing job state, PoI validity, signature authenticity, and double-spend replay prevention).

11.2 Deterministic Mainnet migration protocol

The adapter incorporates a fail-closed network configuration switch between testnet and mainnet. The protocol enforces the following parameters:

- **Mainnet Chain ID:** 5042 (<https://rpc.mainnet.arc.io>, Explorer: <https://explorer.arc.io>).
- **Verified Mainnet Registry Contracts:**
 - IdentityRegistry (ERC-8004): 0x8004A169FB4a3325136EB29fA0ceB6D2e539a432
 - ReputationRegistry (ERC-8004): 0x8004BAa17C55a88189AE136b182e5fdA19dE9b63
 - ValidationRegistry (ERC-8004): 0x8004Cc8439f36fd5F9F049D9fF86523Df6dAAB58
 - USDC Contract Interface (6 decimals): 0x3600
- **Dual-Precision Decimal Handling:** Native gas accounting on Arc uses 18 decimals, whereas the standard USDC ERC-20 contract interface uses 6 decimals (10^6 units per USDC). The adapter enforces exact integer conversions ($u_6 = \lfloor a_{18}/10^{12} \rfloor$) to prevent rounding arbitrage.

12 Requirements for independent reproduction

Independent verification of this paper requires no reliance on proprietary models or unreleased infrastructure. The reproduction package provides:

- the standalone Python simulation harness with deterministic seeds;
- structured JSON and CSV raw outputs;
- vector/bitmap figures directly rendered from data;
- an interactive HTML visualization for browser-based inspection;
- unit and regression test suites;
- cryptographic SHA-256 manifests of all source artifacts; and
- this complete, self-contained LaTeX manuscript.

13 Open problems

Several theoretical and empirical directions remain open:

1. **Dynamic Online Scheduling:** Extending Theorem 1 to dynamic task arrivals, stochastic service durations, and online graph re-coloring, with provable competitive-ratio bounds against an offline optimum.
2. **Fluid Network Stability:** Establishing Lyapunov stability criteria for multi-stage graph-fluid networks under state-dependent gating and bursty arrival distributions.
3. **Standardized Multi-Agent Benchmarks:** Conducting multi-provider empirical evaluations comparing Meru-color scheduling against standard FIFO and priority queues across diverse real-world agent tasks, measuring latency distributions, GPU memory utilization, and cost efficiency.

14 Conclusion

The combinatorial constructions were formally specified and computationally verified, while the current implementation and hardware constraints were evaluated under the observed deployment conditions. By synthesizing binary ranking, Hamming cohorts, graph coloring, conservative fluid transport, evidence gates, and orthogonal telemetry, the system provides provable conflict exclusion and rigorous auditability. The mathematical theorems and conservation invariants are verified independently of any underlying model. The accompanying artifact package ensures complete reproducibility, establishing a transparent foundation for subsequent theoretical extensions and empirical evaluations.

SCIENTIFIC STATUS

Mathematical constructions:
FORMALLY SPECIFIED

Binary ranking:
ANALYTICALLY PROVABLE + COMPUTATIONALLY VERIFIED

Hamming partition:
ANALYTICALLY PROVABLE + COMPUTATIONALLY VERIFIED

Conflict structure:
IMPLEMENTATION INVARIANT - formal graph model under review

Scheduling:
FORMALLY DEFINED + COMPUTATIONALLY VERIFIED
Optimality:
NOT VERIFIED

WIP balance:
ENGINEERING / WORKFLOW CONSERVATION MODEL

Navier-Stokes equivalence:
NOT ESTABLISHED

Hash-chain continuity:
COMPUTATIONALLY VERIFIED

Treasury/accounting conservation:
COMPUTATIONALLY VERIFIED FOR TESTED STATES

LLM/LoRA:
APPLICATION / MEASUREMENT LAYER

Production optimality:
NOT VERIFIED

General superiority:
NOT VERIFIED

References

- [1] Alfonso Amayuelas et al. Self-resource allocation in multi-agent llm systems. *arXiv preprint arXiv:2504.02051*, 2025.

- [2] Amulya Kumar Bag. Binomial theorem in ancient india. *Indian Journal of History of Science*, 1(1):68–74, 1966.
- [3] Jinyuan Chen et al. Kairos: Low-latency multi-agent serving with shared llms and excessive loads in the public cloud. *arXiv preprint arXiv:2508.06948*, 2025.
- [4] Jayant Shah. A history of pingala’s combinatorics. *Resonance*, 2016. Historical analysis of Pingala, Halayudha, and Meru-prastara attribution.
- [5] Barend van Nooten. Binary numbers in indian antiquity. *Journal of Indian Philosophy*, 21(1):31–50, 1993.
- [6] Jinghao Wang et al. Latency-aware orchestration for multi-agent llm workflows on heterogeneous gpus. *arXiv preprint arXiv:2609.03335*, 2026.
- [7] Jinghao Wang et al. Maestro: Workload-aware cross-cluster scheduling for llm-based multi-agent systems. *arXiv preprint arXiv:2606.12950*, 2026.
- [8] Jun Wang et al. Mixture-of-agents enhances large language model capabilities. *arXiv preprint arXiv:2406.04692*, 2024.
- [9] G.-I. Yu et al. Orca: A distributed serving system for transformer-based generative models. In *16th USENIX Symposium on Operating Systems Design and Implementation (OSDI 22)*, pages 521–538, 2022.