PROVISIONAL PATENT APPLICATION — DRAFT (Umbrella)

Title: Systems and Methods for Plan-Based Orchestration, Lattice-Structured Memory, Topological Feature Encoding, Deterministic Compression Verification, and Hardware-Aware Simulation for Reliable Automated Reasoning and Tool Use

> **Disclaimer (not legal advice):** This is a technical draft suitable for a U.S. provisional filing. Provisional applications do not require formal claims, but this draft includes sample claims to set scope. Red-flag items to keep as **trade secret** are clearly marked. Replace bracketed placeholders before filing.

---

1. Inventorship & Filing Metadata

- **Inventor(s):** [Full legal name(s)]
- **Assignee (if any):** [Company]
- **Correspondence:** [Address, email, phone]
- **Attorney Docket:** [Optional]
- **Government Support:** None.
- **Cross-Reference to Related Applications:** None.
- **Sequence Listing:** Not applicable.

---

2. Field of the Invention

Artificial intelligence; program orchestration; information retrieval; signal and text analysis; lossless verification of data transforms; specialized and simulated computing architectures for operator graphs; human-in-the-loop monitoring metrics.

---

3. Background

Large language model (LLM) agents are verbose and probabilistic, which can cause hallucinations, unpredictable latency/cost, and weak auditability. Vector-only memory struggles with structural queries and scale. Hardware sizing for novel operator mixes is typically ad-hoc. There is a need for a **deterministic, budget-bounded controller**, a **structure-aware memory**, a **verified artifact pipeline**, and a **hardware simulation path** usable *before* committing to silicon, with **quantitative self-monitoring** for safety.

---

4. Summary of the Invention

The invention comprises:

1. A **Weyl-State-Machine (WSM)** plan controller that converts a task into an executable operator graph with explicit budgets and a self-check score, producing deterministic, replayable runs.
2. A **Lattice Memory (LM)** that stores artifacts as nodes keyed by content address and **topological/ structural features**, supporting structural retrieval in addition to numeric similarity.
3. A **Topological Feature Encoder** (e.g., Hodge/Hyperkähler-inspired coordinates) and a **K-periodicity detector** for sequences/time-series, enabling structure-aware queries (e.g., "retrieve entries exhibiting near-period K within tolerance").
4. A **Deterministic Compression Verifier** with pluggable `encode`/`decode` that requires exact round-trip; returns ratio and a proof hash for audits.
5. A **Chip Simulator (ChipSim)** that executes the same operator graph under a cycle-level model and reports throughput (IPC) and energy per operation (J/op) for hardware budgeting.
6. An optional **Qualia/Phenomenology Monitor** that reports *measurable proxies only* (broadcast breadth, Brier calibration, integration window, self-consistency) for system health.
7. **Recursive tools/loops** to re-plan based on verifier and simulator results under explicit stopping criteria.

The components expose stable JSON contracts for easy integration.

---

5. Brief Description of the Drawings

- **FIG. 1** — System architecture: WSM ↔ LM ↔ Feature Encoder ↔ Verifier ↔ ChipSim; optional Qualia Monitor.
- **FIG. 2** — Lattice node schema; write and query flows with deduplication via content address.
- **FIG. 3** — Topological feature pipeline from raw input → feature vector → structural key; K-periodicity detection.
- **FIG. 4** — WSM plan lifecycle with budgets and self-check; replay diagram.
- **FIG. 5** — Operator graph mapped to pipeline buffers/stages in ChipSim; IPC and energy reporting.
- **FIG. 6** — Verification loop: round-trip check, proof hash logging, and fail-fast routing.
- **FIG. 7** — Qualia proxy metrics collection paths (broadcast, Brier, retention window, consistency).

---

6. Definitions

- **Operator Graph / Plan:** ordered multiset of named operators with parameters.
- **Budget:** constraints on wall-time, token/compute cost, and memory footprint.
- **Lattice Node:** tuple `(key, payload, features)` with deterministic `node_id = H(key,payload)`.
- **Topological Features:** coordinates derived from manifold-based encodings (e.g., Hodge components, Hyperkähler charts) or equivalent structural encoders.
- **K-Periodicity:** presence of a dominant period near configurable `K` within tolerance `τ`.

- **Proof Hash:** cryptographic digest over raw, encoded, and decoded bytes evidencing round-trip identity.

---

7. Detailed Description of Preferred Embodiments

7.1 Weyl-State-Machine (WSM) Controller

Inputs: task description, policy budgets, optional seeds.
Outputs: `{plan:{ops[], params, seed}, self_check:{score,threshold,explanation}, route, metrics}`.

Algorithm (one embodiment):

1. **Feature query:** compute `qfeat = goal_to_features(goal)` (deterministic embedding; e.g., SHA-based slice to ℝ^d).
2. **Retrieve frontier:** `LM.query(qfeat, k)` for relevant prior nodes.
3. **Synthesize plan:** choose operator sequence from a catalog; include verification and logging operators.
4. **Execute:** call operators; enforce budgets via wall-clock checks and op quotas.
5. **Self-check:** combine verifier result(s) and context presence into a scalar score; set `route ∈ {accepted,fallback}`.
6. **Commit:** `LM.write(key={goal,hash}, payload=artifact, features=artifact_features)`.
7. **Simulate (optional):** pass `plan` to ChipSim to obtain IPC and J/op.

Interfaces:

```
POST /wsm/step
{ "state_id?": "…", "goal": "…", "budget_ms": 250 }
→ { "route": "accepted|fallback", "plan": {"ops":[],"params":{},"seed":"…"},
    "self_check": {"score":0.0,"threshold":0.5,"explanation":"…"},
    "metrics": {"latency_ms":0.0,"cost_usd":0.0}, "chip?": {"ipc":…,
"joules_per_op":…} }
```

7.2 Lattice Memory (LM)

Data model:

- `node_id = SHA256(JSON({key,payload}))` for idempotent dedup.
- `features ∈ℝ^d` (dense); alternative sparse or structural keys permitted.

Operations:

```
POST /lm/write → {"node_id":"…","dedup":true|false}
POST /lm/query → {"hits":[{"node_id":"…","score":0.0}]}
```

Scoring: cosine on aligned features in one embodiment; structural poset-based similarity in another (meet/join distance over lattice labels).

7.3 Topological Feature Encoder & K-Periodicity Detector

Encoder (embodiment): map inputs (symbolic sequences or time-series) to coordinates via a differential-geometric procedure (e.g., Hodge decomposition of derived series, projection to a Hyperkähler-like chart), then normalize to a fixed vector.

Periodicity detector: given a positional series `y`, compute `Y = y − mean(y)`; let `P = |FFT(Y)|^2`; infer periods by `T = 1/f` for `f>0`; report peaks near configurable `K` within tolerance `τ` and strength `S`.

Interfaces (optional):

```
POST /nlp/analyze     → topological tokens / features for symbolic inputs
POST /sta/analyze     → features for numeric series; returns {area, peak,
fwhm, duration, mu, tokens}
```

7.4 Deterministic Compression Verifier

Contract:

- `encode(raw:bytes) → bytes`, `decode(packed:bytes) → bytes`.
- **Requirement:** `decode(encode(raw)) == raw` (bit-exact) for all artifacts prior to acceptance.
- **Report:** `{ratio = |enc|/|raw|, ok_roundtrip, proof_hash = H(raw||enc||dec)}`.

Interface: `POST /comp/verify` with base64 payload.

> **Trade secret boundary:** keep the prime-based codec internals proprietary if desired; the verifier API remains public.

7.5 Chip Simulator (ChipSim)

Inputs: `op_graph` (the plan) and `cycles`. **Model:** per-operator cost table; pipeline utilization model; energy model yielding `J/op`. **Outputs:** `{ipc, joules_per_op, trace_id}`.

Interface: `POST /chip/run`.

7.6 Qualia/Phenomenology Proxies (Optional)

Metrics:

- **Broadcast breadth:** fraction of subsystems receiving global messages.
- **Brier calibration:** mean squared error of self-reported probabilities vs outcomes.
- **Integration window:** retention metric across delay.
- **Self-consistency:** contradiction rate over identity/state prompts.

Interface: `POST /pheno/ping → {report, proxies:{broadcast, integration_ms, brier, self_consistency}}`.

7.7 Recursive Tools/Loops

WSM may re-enter with updated budgets or alternate plans based on verifier/simulator outcomes until acceptance or budget exhaustion. Stopping criteria and backoff strategies are parameterized.

---

8. Example Embodiments (Non-limiting)

1. **Symbolic-sequence audit:** controller plans {retrieve→analyze→verify→log}, uses LM to fetch prior analyses with similar topological features, verifies artifacts by round-trip compression, and commits with proof hash.
2. **Time-series screening:** STA analyzer computes {area, peak, FWHM, duration, μ}; K-periodicity check flags near-K patterns; WSM logs pass/fail and triggers ChipSim to budget edge deployment.
3. **Edge sizing:** with a stable operator mix, ChipSim reports IPC and J/op; WSM selects a plan variant meeting target energy/latency.

---

9. Advantages

- Deterministic, budgeted execution with replay.
- Structure-aware retrieval beyond vector cosine.
- Built-in verification (compression round-trip) for artifact integrity.
- Hardware-aware metrics (IPC, J/op) before fabrication.
- Quantitative self-monitoring for safety.

---

10. Alternative Embodiments

- Replace cosine with graph kernels or lattice meet/join distance.
- Replace FFT with autocorrelation, cepstrum, or wavelet-based periodicity.
- Swap prime-based codec with any deterministic, reversible transform family.
- Swap ChipSim model with dataflow/NoC or FPGA-accurate timing.

---

11. Informal Claims (Provisional friendly)

Claim 1 (Method). A method comprising: (a) receiving a task; (b) generating, by a controller, an operator graph with budgets; (c) executing the graph to produce artifacts; (d) encoding inputs and artifacts into topological feature vectors; (e) storing artifacts in a lattice-structured memory with content address and features; (f) retrieving prior artifacts via structural similarity including K-periodicity; (g) verifying artifacts by deterministic round-trip compression with proof hashing; and (h) simulating the operator graph on a hardware model to obtain throughput and energy metrics for decision-making.

Claim 2 (System). A system comprising modules for the method of Claim 1: a plan controller (WSM), a lattice memory (LM), a topological encoder with K-periodicity detector, a compression verifier, a chip simulator, and an optional phenomen