A11 — Architectural Invariants

Canonical, Non-Negotiable Structural Rules of the A11 Architecture

Version 1.0

Aleksej Dvojnev

Abstract

This document defines the *architectural invariants* of Algorithm 11 (A11), the universal deterministic reasoning architecture. These invariants represent the non-negotiable structural, geometric, and logical properties that must be preserved in any compliant implementation of A11. They ensure that the architecture remains stable, interpretable, and reproducible across engineering, software, and autonomous-system contexts. The invariants formalize the dual-branch structure of Wisdom and Knowledge, the role of Comprehension as the sole integration node, the deterministic operational cycle (L5–L11), mandatory constraint gates, rollback mechanisms, bounded recursion, and strict level ordering. Any system that violates these invariants is not considered an implementation of A11, regardless of terminology or partial similarity. This document serves as a protective layer for the architecture and must be referenced before creating demonstration cases, reference implementations, or system integrations.

Table of Contents

0. Purpose of This Document

This document defines the **architectural invariants of Algorithm 11 (A11)** — the properties that are:

- **immutable**,
- **mandatory**,
- **non-negotiable**,
- **not subject to interpretation**,
- **not allowed to be simplified**,
- **not allowed to be removed**,
- **not allowed to be replaced by implementation details**.

Any system that violates even one invariant is **not an implementation of A11**, even if it uses similar terminology or level structure.

This document protects the A11 architecture from distortion during engineering demonstrations, reference implementations, and system integrations.

1. Geometric Invariants

1.1 Dual-Branch Structure (Wisdom / Knowledge)

- Levels **Wisdom (L2)** and **Knowledge (L3)** must always operate **in parallel**, never sequentially.
- These levels form a **dual-branch weighting structure**, not a linear pipeline.
- No implementation may convert Wisdom and Knowledge into a sequential process.

1.2 Comprehension as the Balancing Node (L4)

- Comprehension is the **sole integration node** between Wisdom and Knowledge.
- Comprehension must **balance both branches**.
- Comprehension is the **only valid entry point** into the operational layer (L5–L11).
- Skipping, replacing, or simplifying Comprehension is prohibited.

1.3 Stable Core (L1–L4)

- Levels L1–L4 form the **stable core** of A11.

- These levels cannot be removed, reordered, merged, or bypassed.
- Any instability in reasoning requires returning to L1–L4.

1.4 Operational Cycle (L5–L11)

- Levels L5–L11 form a **deterministic operational cycle** that always follows Comprehension.
- The order of L5–L11 is fixed and cannot be altered.
- No level may be removed, merged, or replaced.

1.5 Bounded Recursion

- All recursion in A11 must be **explicitly bounded**.
- Infinite recursion is prohibited.
- Recursion depth must be defined or computable.

1.6 Semantic Branching

- Semantic branching is mandatory at L3 and L5.
- Branches must be explicit, not hidden inside implementation logic.

2. Logical Invariants

2.1 Deterministic Transitions

- Transitions between levels must always be **deterministic**.
- Probabilistic transitions between levels are prohibited.

2.2 Mandatory Constraint Gates

- Constraint checks are mandatory at L5 and L7.
- Constraint checks must produce a binary result: **pass / fail**.
- Skipping constraint checks is prohibited.

2.3 Mandatory Rollback

- When a conflict, contradiction, or constraint violation occurs, the system must perform a **rollback**.

- Rollback is not optional.
- Rollback must return reasoning to the nearest stable point (typically L4).

2.4 Single Integration Point

- Comprehension is the **only** integration point in the architecture.
- No other level may perform integration of branches.

2.5 No Level Skipping

- No level may be skipped.
- No levels may be merged.
- No levels may be executed out of order.

2.6 Deterministic Reasoning Trace

- A11 MUST generate a complete, deterministic reasoning trace for every decision cycle.
- The trace must include:
- • semantic branches (L3, L5)
- • evaluation results (L6)
- • constraint gate outcomes (L7)
- • rollback triggers and restored states
- • final decision justification
- • the context frame used for the cycle
- The reasoning trace must be:
- • deterministic — identical inputs produce identical traces
- • auditable — every step can be inspected
- • verifiable — omissions and contradictions must be visible
- • replayable — the entire cycle must be reproducible offline
- Absence of a reasoning trace is a violation of A11 invariants.

3. Structural Invariants

3.1 Separation of Concerns

Each level of A11 has a strictly defined function.

Functions of one level may not be moved to another.

3.2 No Self-Modification

A11 cannot modify its own architecture.

Implementations may not:

- add levels,
- remove levels,
- change level meaning,
- change level order.

3.3 Deterministic Cycle Completion

The operational cycle (L5–L11) must always terminate.

Cycles without a defined exit are prohibited.

3.4 Stability First

If reasoning becomes:

- contradictory,
- unstable,
- incomplete,
- ambiguous,

the system must return to L1–L4.

4. Forbidden Modifications (Non-Compliance Conditions)

Any system exhibiting any of the following is **not A11**.

4.1 Sequential Wisdom → Knowledge

Wisdom and Knowledge may not be executed sequentially.

4.2 Missing Comprehension

Comprehension may not be skipped, replaced, or embedded elsewhere.

4.3 No Rollback

Ignoring rollback during conflict is prohibited.

4.4 No Constraint Checks

Executing the operational cycle without constraint gates is prohibited.

4.5 Infinite Recursion

Unbounded recursion is prohibited.

4.6 Level Merging

Merging levels (e.g., L2+L3 or L5+L6) is prohibited.

4.7 Level Reordering

Reordering levels is prohibited.

4.8 Hidden Branching

Semantic branching may not be hidden inside implementation logic.

4.9 Probabilistic Transitions

Probabilistic transitions between levels are prohibited.

4.10 Partial-Use of A11

Any implementation that executes only selected levels of A11, skips Comprehension, omits constraint gates, bypasses rollback, or suppresses reasoning trace generation is **not A11**.

Partial execution violates the architectural invariants and invalidates compliance.

5. Compliance Requirements

Any A11 implementation — engineering, software, simulation, or architectural — must:

- adhere to all invariants,
- document branching points,
- document constraint checks,
- document rollback mechanisms,
- document transitions between levels.
- generate and expose a deterministic reasoning trace for every cycle,
- execute all 11 levels without omission,
- partial execution is strictly prohibited.

Violation of any invariant means the system is **not A11**, regardless of terminology used.

6. Role of Invariants in Demonstration Cases

This document is mandatory before creating:

- engineering demonstration cases,
- reference implementations,
- Python models,
- architectural integrations.

It ensures that:

- A11 remains primary,
- implementation does not dictate architecture,
- demonstrations validate A11,
- A11 does not become an “interpretation” or “variant”.

References

- A11 Structural Architecture Specification
- A11 Cognitive Architecture
- A11 Decision Layer
- A11 Language Specification