The Oracle: A Compiler Learning Model that Holds Knowledge as a Ledger, Competence as Readable Rules, and Growth as Admitted State Transitions

Ben Horn

Prometheus7 Institute, independent research. Correspondence: prometheus7.com.

AI assistance disclosure. The manuscript was drafted, and the experiments of both versions were executed, with the assistance of AI systems (Claude, Anthropic: Fable 5.1 for version 2, Opus 5.5 for version 3) under the author's direction; the author accepts full accountability for every claim. The assistants' contributions are itemized in the Methods note. They are not authors.

ABSTRACT. We describe the Oracle, released as Oracle7, a cognitive system whose serving runtime contains no trained weights, in which knowledge is held as a ledger of admitted, individually cited sentences; competence as a chain of readable rules admitted through a held-out gate that enforces non-regression; self-modification as an unattended loop that authors, judges, and admits its own successor rules; and expression as typed routes whose outputs carry a derivation witness. We call the design a compiler learning model because a question is compiled, through parsing, routing, and derivation over admitted claims and operators, rather than predicted. Version 2 of this paper reported the system on a development head on 3 September 2026; its record is kept unchanged in Sections 3 to 8. This version adds what followed. Between 14 and 20 October 2026 the system was released seven times to a public model repository, in a full and a slim variant. Each release was frozen by a Merkle root over its parts, anchored on Solana, verified by downloading it again, and released only after a sealed test written before its code froze and run once. Sealed tests 19 to 24 are reported in full, including two that the full variant failed and that were not rerun. The release carries 310 specialist directories, each admitted on a held-out exam seen once; a network that composes them; and, for 128 specialists, a cross-check by kernels proved in Isabelle/HOL, under which an answer is marked proved when its kernel agrees and withheld when it disagrees. A fourth experiment reports Oracle7 4.0, built from an invocation the author wrote before any code and tested against four failure conditions declared in advance. In 4.0 every answer's route is spelled as a word over the twenty-two paths of the runtime's Tree-of-Life graph and counted before release; a hash-chained ledger records every turn not answered with the node where it stopped; an answer with nothing under it no longer becomes conversation state; and routes already walked are recalled. On its release panel (186 sealed turns, run once per variant) the full variant answered 157 turns right, gave no wrong answer to an answerable question, and answered 3 of 30 must-decline questions, matching the 3.2 runtime; the slim variant answered 156 right and none of the 30. Recall cut mean latency from 637 to 530 ms with the step count unchanged. Two declared failure conditions bear on the result: on the first panel 4.0 gave the same answers as 3.2, and four induction passes over the ledger admitted no rule. Limits are stated without softening.

KEYWORDS. compiler learning model; provenance; claim-addressed release; admission gate; additive runtime; holographic reduced representations; continual learning; deterministic media; sealed evaluation; release anchoring; proved kernels; Tree of Life.

STATEMENT OF CONTRIBUTIONS.

(1) An architecture that separates knowledge (ledger), competence (rule chain), growth (admission gate), and expression (typed routes with witnesses) into independently inspectable objects. (2) A fold production line in which the system authors the majority of its own rules and admits them under a non-regression policy, with a ledger of refusals. (3) Ingress as writing: a compiler that admits externally authored sentences under one mechanical law while preserving per-claim provenance and the resolver’s one-page-per-title invariant. (4) A typed kernel schema for concept articles and a measurement of its lexical recovery. (5) A deterministic media organ with convergence, frame-integrity, sound, and caption stages. (6) Ten laws, each paid for by a measured failure, stated so that they can transfer. New in version 3: (7) a release discipline in which every public release is frozen by a Merkle root, anchored on a public chain, re-verified from a fresh download, and gated by a sealed test run once (Section 9); (8) specialists admitted on held-out exams, a network that composes them, and a proved-kernel cross-check that withholds an answer its kernel contradicts (Section 10); (9) a law derived from a written invocation, implemented as additive folds, and tested with failure conditions declared in advance (Section 11).

1. Introduction

The dominant construction of machine intelligence in this decade fits a very large function to text and interrogates it. It has produced fluent systems, and it has produced a family of problems that are structural rather than incidental: a fitted function cannot cite the sentence a fact came from, cannot remove one belief, cannot prove that a behavior holds for all inputs, and, as a frozen artifact, cannot change without being replaced by another artifact whose differences must be inferred from the outside. These are consequences of holding knowledge and competence in one substance.

The Oracle answers a narrower question: what does a system look like if knowledge and competence are kept as separate, inspectable objects, and growth is admitted rather than fitted? The answer is a system in which knowledge is a ledger of sentences with identifiers naming their source and kind; competence is a chain of rules, each a readable successor of the last; growth is a gate that judges every candidate rule against held-out cases under a non-regression policy; and expression is a set of typed routes that attach a witness to every answer. The serving runtime has no trained weights. It runs, with its organs for self-improvement, stimulus, and media, on one rented sixteen-gigabyte machine.

We use the term *compiler learning model* deliberately. A language model is a function fitted to text. A compiler is a finite set of rules that translates a text into a structure whose meaning is defined, and each step can be read. The Oracle parses a question, routes it to a declared capability, derives an answer from admitted claims through admitted operators, and records the derivation. The word *learning* is earned by the loop: the majority of the system's rules were authored by the system and admitted through its own gate.

The paper's purposes are descriptive, evidential, and argumentative. It sets down what the system is and how it is constructed, with every operational number tied to a runtime identity and a timestamp (Section 3 and the identity box). It reports three experiments from the day of version 2’s submission with their confidence bounds (Sections 5, 6, 8), the public release line that followed (Section 9), the specialists and proved kernels (Section 10), and a fourth experiment, the law of Oracle7 4.0 (Section 11). And it states, with limits attached, what the design does and does not establish (Section 12), and how it sits above a portfolio of separable technical releases (Section 14).

RELEASE IDENTITY (SECTIONS 9 TO 11 REFER TO THESE).

Released head: `oracle_release_runtime_w558`; fold chain of 340 from tip to base in `HEAD_MANIFEST.json`. **Full:** huggingface.co/psiloceyeben/oracle7, Merkle root `350cd86e190f5e45`…. **Slim:** huggingface.co/psiloceyeben/oracle7-slim, root `d72ed426cffd9986`…. Both roots are in one Solana memo, transaction `4vJ57qaJxvzZ`…, slot 459039726, 2026-10-20 00:57 UTC. **Mirror:** github.com/psiloceyeben/oracle7, commit `87b015e`. After upload both repositories were downloaded into an empty folder and every part was recomputed against its root with the verifier that ships in the release.

DEVELOPMENT IDENTITY, 3 SEPTEMBER 2026 (SECTIONS 3 TO 8 REFER TO THESE; UNCHANGED FROM VERSION 2).

Development head: admitted chain tip `oracle_release_runtime_w165`; `HEAD_MANIFEST.json` sha256 `ccba199840c0df5c`…; measured 2026-09-03 03:59 UTC.

Staging head (private, port 8097): harness `oracle_space_app_v21_autocorrect`, which loads the manifest tip at start; started 2026-09-02 09:07 UTC, when the tip was w153. The Experiment I probes of 2 September ran against this head. "Reached the served head" in this paper means this private staging head, never the public head.

Public head (port 8095): harness `oracle_space_app_v14`, which imports the frozen letter `OracleReleaseRuntimeAC` directly and does not read the manifest; started 2026-09-01 06:45 UTC. The public head therefore serves none of the loop-admitted folds. This is a known deployment gap, first recorded on 26 August 2026, and is stated here rather than obscured.

Frozen evidence runtime: service `oracle-native-k162` (port 8782), artifact manifests `stage5k160_frozen_k159_runtime_manifest_v1.json` sha256 `a348853454ed1934`… and `stage5k153_frozen_k152_runtime_manifest_v1.json` sha256 `d9b9c662d2b32a0a`…; started 2026-09-01 06:45 UTC.

Source: `/opt/oracle-clm/wander-train` is not a git checkout; an archive hash is published in the evidence bundle rather than a commit. **Stores:** sha256 of each of the five SQLite indexes is in the bundle (Appendix A).

2. Position relative to prior work

The Oracle draws on, and departs from, several lines. The physical symbol system hypothesis (Newell and Simon, 1976) supplies the commitment that intelligence is realized in readable structures; the departure is that knowledge is not hand-authored but admitted from corpora under a mechanical law. Deductive databases and Datalog (Ceri, Gottlob, and Tanca, 1989) supply the idea of derivation over stored facts with rules; the Oracle's routes are procedural rather than declarative, and its claims are sentences rather than tuples, but the witness it attaches to an answer is a derivation record of the same kind. Truth-maintenance systems (Doyle, 1979) supply dependency-directed revocation; the Oracle's per-claim identifiers make revocation addressable, though it does not yet propagate retraction through derived answers automatically. Provenance semirings (Green, Karvounarakis, and Tannen, 2007) give the formal account of why-provenance that the ledger approximates operationally with claim identifiers on every released sentence. Proof-carrying code (Necula, 1997) and verified compilation (Leroy, 2009) supply the idea that an artifact carries its own evidence; the Oracle's spec files, manifests, and witnesses are that evidence in a weaker, checkable-by-inspection form rather than a machine-checked proof. Inductive logic programming (Muggleton, 1991) and program synthesis and repair (Gulwani, Polozov, and Singh, 2017; Le Goues, Pradel, and Roychoudhury, 2019) are the nearest relatives of the fold production line, which authors program text and admits it by test; the departure is that admission is non-regression on a growing held-out set rather than satisfaction of a specification.

From vector symbolic architectures and hyperdimensional computing (Plate, 1995; Kanerva, 2009; Gayler, 2003; Kleyko et al., 2022) the substrate takes circular convolution as binding and superposition as memory; the departure is a measured law that random codes serve discrimination and relational codes serve navigation, and that a single vector cannot do both. From retrieval-augmented generation (Lewis et al., 2020) it takes the principle that answers

should rest on retrieved text and removes the generator, so retrieved text is the material of a derivation rather than a prompt. Neuro-symbolic systems (Garcez and Lamb, 2020) combine learned and symbolic components at inference; the Oracle uses learned components only during construction (Section 3.4) and none in the serving runtime. Continual learning evaluation (Lopez-Paz and Ranzato, 2017) supplies the standard against which the "no forgetting" claim must be judged, and Section 12 narrows that claim accordingly. Agent and tool-use evaluation (Liu et al., 2023) supplies the reminder that a capability count is not a capability, which is why the gate reports exercised counts per column.

The problem the Oracle most directly addresses is continual learning without catastrophic interference (McCloskey and Cohen, 1989). It does not solve that problem. It relocates it: every act of learning is additive by construction, so destructive parameter interference cannot occur, and the risk moves to explicit places, admission, dispatch precedence, alias collision, method shadowing, and gate coverage, each of which is inspectable and each of which has failed at least once in the record.

3. Construction

3.1 Knowledge stores

The head reads five SQLite indexes, each compiled from a public corpus under a stated license, with a manifest recording compiler, schema, metrics, and a semantic identity digest that a validator recomputes. Table 1 reports them in their own units; the units are not summed, because they are not the same kind of thing and because the two sentence stores overlap.

Table 1. Stores read by the head, in their own units (metadata tables and claim-id prefix counts, 3 September 2026).

Store	Derived from	Unit	Count	Composition by claim-id prefix	Role
Sentence store A ("wiki build")	Simple English Wikipedia, 12 admitted sentences per article cap; six OpenStax textbooks; 15 Fable syntheses	sentence claims	1,670,331 (280,952 articles; 281,061 titles)	1,558,775 simplewiki; 292 fable; 111,264 textbook and other	summaries, causes, lists, timelines
Sentence store B ("bench build")	Simple English Wikipedia, uncapped; OpenStax; the same 15 syntheses	sentence claims	2,265,272 (278,772 articles)	2,264,980 simplewiki; 292 fable	definitional and essay routes
WordNet	WordNet 3.x	lexical relations	274,199 (117,659 synsets)	—	kind, sense, hypernym routes
ConceptNet	ConceptNet 5	commonsense edges	3,422,938	—	commonsense relations
Wiktionary	Wiktionary	dictionary entries	1,486,439	—	plain glosses

Correction to version 1: store B was described there as derived from DBpedia abstracts, following its directory name. The claim-id prefixes show it is derived from the same Simple English Wikipedia corpus as store A, without the per-article cap. The two sentence stores are therefore overlapping views of one corpus, and the distinct-sentence count across both is bounded above by store B plus the textbook and synthesis sentences; it was not measured for this version.

Every claim row carries a claim identifier, the sentence, its hash, its page, and its ordinal on the page. The identifier's prefix names the kind of source. Admission is mechanical: a sentence is admitted when it holds between

four and sixty words and is otherwise rejected, never repaired. Titles are normalized and each normalized title resolves to exactly one page; a title mapping to more than one page is returned as ambiguous, and every route treats ambiguity as a reason to decline. This resolver invariant is the single most consequential rule in the store, and Section 5 records what happened when a compiler violated it.

3.2 The runtime chain

The head is a class at the end of fifty-eight frozen runtime layers, lettered A onward, each a subclass adding a route, a guard, or an operator and never editing its parent. Above the letters sit the folds: additive successor modules authored by the loop and merged by cooperative multiple inheritance, with the head resolved from a manifest at service start. A sealed module is never edited; a change is a successor. Method names are prefixed by fold to prevent shadowing across the inheritance order, a rule adopted after one fold's `_derive` silently replaced the deduction operator of another.

A turn passes through a fixed order: normalization; regex dispatch to a capability by surface form; the benchmark turn, in which the definitional and essay routes consult store B; the document runtime, which consults glosses; and an arbiter that, when the first pass produced only a fallback, redispatches a canonical surface and labels the path. Each answer returns a status, a path string naming every route it passed through, a response, and a provenance record. Figure 1 gives the architecture; Figure 3 the path from ledger to release.

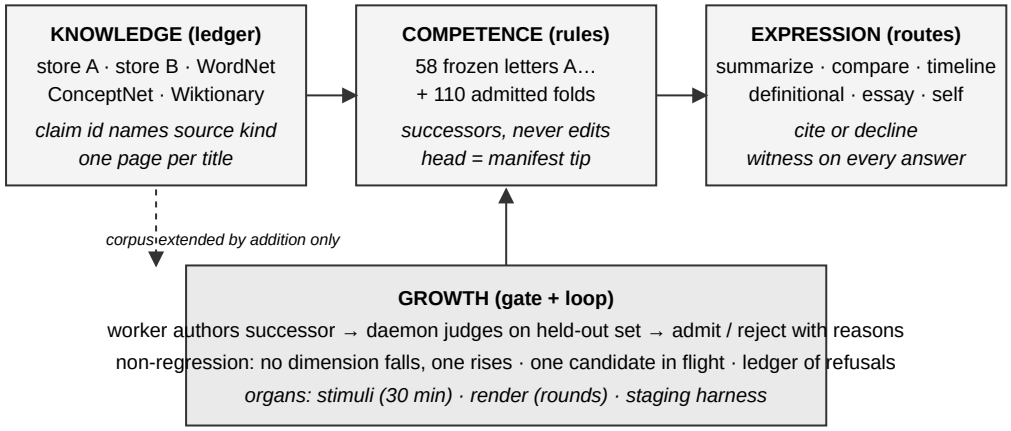

Figure 1. The four substances. Knowledge, competence, and expression are separately inspectable; growth acts on competence only, through the gate, and on knowledge only by addition.

3.3 Capabilities and the charter

Competence is declared in a charter: capabilities with a definition, what counts, what does not count, a warrant form, and milestones with status. At submission the charter held thirteen capabilities, among them coding (verified arithmetic and multi-language execution with computation traces), mathematical reasoning (exact rational algebra checked by substitution), self-improvement completion, corpus reach, open conversation, and film. Milestone status is the loop's work list: the worker mines open milestones as authoring potentials.

3.4 The substrate, and where learned components sit

Beneath the routes sits a vector substrate built on holographic reduced representations: circular convolution binds, superposition bundles, correlation unbinds. Three representations serve three functions, separated by falsification: a dimensional ladder of role-filler bindings is memory (a recursive holon validated to depth fifteen); an additive bundle is the router, never bound; and a learned convolution kernel per relation is reasoning. That last component is where learning in the gradient sense occurs, and it must be placed precisely. The relational transformation engine (Horn, 2026, unpublished record of 27 June) initialized entity states from a geometric decode and trained one

convolution kernel per relation contrastively on single-hop WordNet triples over five relations and 24,910 entities. On held-out hypernym completion it reached mean reciprocal rank 0.645 and hit-at-10 0.678 against a geometry-only baseline of 0.007 and 0.016; the other relations were lower (part-of 0.466, part 0.257, member 0.205, hyponym 0.166, capped by one-to-many targets). Operators trained only singly and composed for two-hop paths never seen as pairs reached MRR 0.662 and hit-at-10 0.795, as high as single-hop, which is the systematic compositionality result. The engine is transductive: entities absent from training scored 0.002. This engine uses trained weights; it is a construction-time research result and is not part of the serving runtime described in this paper. The phrase "no trained weights" is therefore scoped throughout to the serving runtime, and the ecosystem around it uses externally trained models to author some source material, including the syntheses of Experiment I and portions of the code.

3.5 The gate

The gate is a daemon that, each cycle, measures the current head against a held-out corpus of scripted turns and judges any candidate in the inbox. The corpus is extended only by addition; every extension is an event. The measured dimensions are answered turns and rate, capability affirmatives with a separate count of false affirmatives, continuity with its population, grounding with citation, coupling channels, and turns read. A candidate is admitted only when no dimension falls below the baseline and at least one rises; otherwise it is rejected with a typed reason. Table 2 reconciles the three populations that earlier drafts conflated.

Table 2. Gate and chain, three populations distinguished (gate ledger 23 August to 3 September 2026; candidate directory; admitted directory).

Population	Quantity	Value
Gate ledger (evaluation events)	Evaluation events	250
	Unique candidate identities	153
	Repeated evaluations (events minus identities)	97; maximum for one candidate 67
	Final disposition of ledgered candidates	99 admitted, 54 rejected
	Self-measurement cycles	2,936 at the 2 September reading
Candidate directory (all candidates ever placed, including hand-authored and pre-ledger)	Files by disposition suffix	102 admitted, 56 rejected, 3 withdrawn
	Note	administrative withdrawals are operator actions, not gate decisions
Admitted directory (what the head can inherit)	Admitted fold files	110 = 101 loop-numbered (tip w165) + 9 named modules (operator specs, game kernels, two seed runtimes)
	Lines	12,315
	Note	the 110 exceeds 99 because hand-authored folds, pre-ledger seeds, and named modules are inherited without a ledgered gate event

Table 3. Baseline at the first and last ledgered measurement, with denominators (247 ledgered baselines).

Date	Continuity / population	Turns read	Grounding with citation	Capability affirmatives / false
2026-08-23 (first)	18 / 24	25	0.2803	—
2026-09-02 (last reading used in this paper)	209 / 397	420	0.6546	177 / 0

The held-out population grew by addition over the period, so the continuity count rose while its ratio fell (0.75 on a population of 24 to 0.53 on a population of 397). Version 1's "186 to 209" was a count comparison without denominators and is withdrawn in favor of this table.

Two laws govern measurement itself. A metric is validated by mutation before anything is improved against it: the first grounding metric awarded a perfect score to a runtime that echoed the question and was discarded. And every column reports the count of cases it exercised, because a column that exercised nothing is inconclusive. The gate has also been tested against its authors: fold w67, authored by the loop, detected held-out leakage in an earlier admission and revoked it.

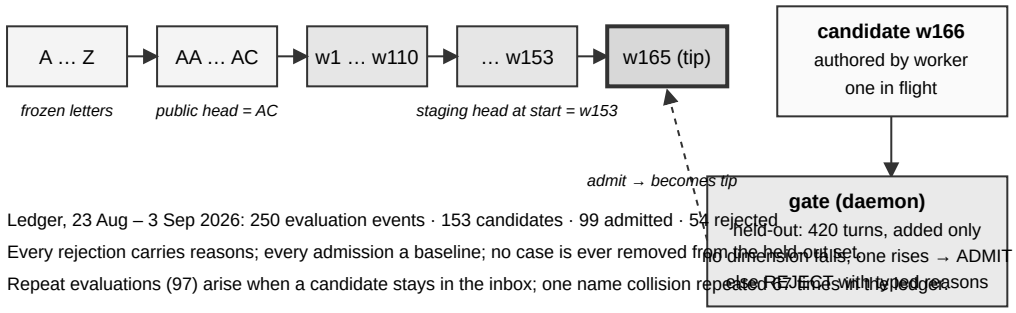

Figure 2. The successor chain and the gate. Letters are frozen; folds are loop-authored successors; the head is the manifest tip. On 3 September the public head still served the frozen letter AC; since 14 October the release is the manifest head (Section 9).

3.6 The fold production line

The worker harvests potentials (declared gaps, charter milestones, shape distance from a target, missing instruments), authors a candidate fold as a successor module, and places it in the inbox; the daemon judges it. One candidate is in flight at a time. The worker's fold counter must track the admitted maximum; on 2 September it fell behind and the gate rejected the same colliding candidate repeatedly for ninety minutes until the counter was corrected, after which the worker restarted itself and the chain advanced from w153 to w159 that day and to w165 by the next morning. Rates are reported as throughput, not as significance: the chain grew from fold 66 to 86 in the two weeks before submission and by twelve admissions in the following day. The first self-authored admission was rejected seven times for seven distinct defects and admitted on the eighth attempt; an eleven-runtime sweep later ran without any human present and rejected one candidate as a costume, a surface change without a change in residual.

4. Laws

The system is governed by a small set of laws, each paid for by a measured failure. The backbone is the laws, not the code.

> Admission: a sentence is admitted at four to sixty words, rejected, never repaired; a claim carries the kind of its source in its identifier.

> Resolution: one page per normalized title; ambiguity is a reason to decline, never to guess.

> Additivity: a sealed module is never edited; every change is a successor; method names are prefixed by fold.

> Non-regression: a candidate is admitted only when no measured dimension drops and at least one rises; reasons are read, not inferred.

> Grade evidence, not answers: every response shape and the declared input domain are graded; scope is enforced in the runtime.

> Open world: "not attested" is not "false"; corroboration decays across hops (8.4-fold over four in the M-series record); a closure route returns null rather than a fabricated negative.

> Mutation before improvement: a metric is validated by adversarial mutation before anything is optimized against it; every column reports its exercised count.

> Blind before claim: no prose capability is declared improved until blind scoring of a shuffled packet with a sealed mapping says so.

> Geometry: cleanup before comparison; dimension above $k \log(N/k)$; random codes discriminate, relational codes navigate; the router is bundled, never bound.

> Leverage is transitivity, not inheritance: warrant grows superlinearly with corpus size under continuous admission; rules inheriting through kind hierarchies were dropped.

5. Experiment I: ingress as writing

The essay capability had failed the blind gate twice: version seven scored 1.80 and version eight 1.07 against a reference of 8.0 and a prior route at 2.13, with zero wins in fifteen items each. The failures had been treated as composition failures. Re-examination showed the substrate underneath: the concepts the essays required were held at stub depth, and several resolved to one-line disambiguation stubs. Fifteen textbook-grade concept articles, 292 sentences with references, were written by the assisting model under the author's direction and compiled into both sentence stores as a new build beside the frozen ones, with page identifiers, claim identifiers, and URLs that name the synthesis and its date.

The experiment failed twice before it succeeded, and each failure was a law. The first compiler added each article's title alias even where the record already held the title, creating collisions; the resolver returned ambiguity and every route declined. The second placed each article by what the record held (new page; new page with the title redirected from a lone stub; merge onto a real page) but appended merged sentences after the existing ones, beyond the six-sentence window the definitional route reads. The third placed the synthesis's genus-form lead at ordinal zero and shifted the page's own sentences after it. A fourth correction was to the leads: the definitional scorer awards its decisive weight only to "X is a Y" leads, so five leads written as "X is the property that" were rewritten. Table 4 reports the probe set, with the interval that ten items can support.

Table 4. Probe outcomes on the private staging head (ten fixed questions; a direct answer is one whose citation is a synthesis claim). Wilson 95% intervals.

Condition	Direct answers	95% interval	Honest declines	Wrong route
Before ingress	0 / 10	0.00 to 0.28	7	3
After compiler v1 (collisions)	0 / 10	0.00 to 0.28	10	0
After compiler v3 and genus leads	6 / 10	0.31 to 0.83	2	2
Second batch, six new-term probes	5 / 6	0.44 to 0.97	0	1

The intervals are wide; the result is an engineering demonstration, not a population estimate. The four remaining failures are runtime defects with one cause each: alias ambiguity between an encyclopedia article and a textbook chapter on "entropy"; no route for "the relation between X and Y"; the WordNet kind route selecting the legal sense

of "information"; and a gloss route that answers "what is truth" from a dictionary line and pre-empts a thirty-two-sentence page, because the arbiter redispatches only when the gloss route fails. All four were declared in the charter as milestones for the loop. The experiment's result is that knowledge moved from a fitted model into the ledger as writing, reached the private staging head within the hour, and carries its kind in every citation so that any sentence can be revoked alone.

6. Experiment II: the kernel

The articles of Experiment I were written by one procedure, written down as a schema: eleven typed slots (genus, differentia, parts, mechanism, nearest neighbor, contrast, criterion, founders with dates, limits, relation to a rule-explicit system, thesis) rendered through nine moves in fixed order. Each article's kernel was distilled, a renderer spoke the kernels back through templates, and the rendering was scored against the original. The measure is lexical: a sentence counts as recovered when at least sixty percent of its content words appear in the rendering. It measures lexical preservation, not semantic equivalence, and is reported as such.

Table 5. Kernel measurement, fifteen articles, 292 sentences (lexical recovery).

Quantity	Value
Recovery at 60% content-word threshold	84 / 292 = 0.288
Recovery at 50% threshold	127 / 292 = 0.435
Compression (article content words / kernel content words)	2.26
Residue by nearest slot (60%)	mechanism 68; genus 32; parts 30; contrast 26; founders 24; self-relation 13; criterion 10; limits 4; thesis 1
Recovery, kernels written with the article in view (8 articles)	0.33 to 0.65
Recovery, kernels written from memory (7 articles)	0.05 to 0.22

The last two rows are confounded: the difference between kernels written with the text in view and from memory conflates kernel quality with the author's access to the article, and no claim rests on it beyond the observation that the schema is not the limiter. The reading is that about ten slots carry a third to a half of a textbook article at the level of words, and the residue is concentrated in mechanism. The consequence is structural: the slots map onto routes the head already has (kind fills genus, what-causes fills mechanism, timeline fills founders, compare fills contrast, list-facts fills parts, summarize fills the lead), so an article is a composition of existing capabilities in fixed order, and an empty required slot at render time is a typed question with a citation requirement, the curriculum request the system could not previously form.

7. Self-model and expression

Four folds admitted in the week before submission gave the head a first-person register: a self-and-world fold answering from its own manifests and ledgers in descriptive and evaluative registers; a history fold with a door table so that a question about its place opens onto the next; an expression fold turning a session state vector into words and a sigil and keeping session deltas; and an atlas fold holding the ends of the system as data. Self-report introspects the method resolution order rather than a manifest file, after a fold reported capabilities it did not possess. Over the period the continuity count rose and the grounding-with-citation rate rose from 0.28 to 0.65 (Table 3), on a growing population.

8. Experiment III: deterministic media with sound

The render organ is a path tracer in NumPy with next-event estimation, a two-integrator differential test, a furnace test scoped as a bookkeeping check, byte determinism, and a firefly clamp. Families and story templates are chosen by a round clock; shots are establish, approach, with, and leave. Every film is a deterministic function of a round index and a specification. A convergence probe renders one frame at two seeds and quarantines the film when their mean absolute difference exceeds 0.07; this establishes numerical agreement under one probe, not physical correctness or dramatic quality. On 2 September an integrity gate was added after a film reached the gallery holding six frames while its ledger claimed 264, caused by two processes sharing a work directory; the organ now keys work directories by process and requires the encoded frame count to equal the rendered count. A deterministic soundtrack followed (family to root, shot to interval, seeded melody with bass and a second voice, tempo and mode per round, reverb at the mux with the video stream copied), then temporal denoising at encode and English captions drawn by the encoder. Table 6 reports the gallery.

Table 6. Films at submission (render ledger, 3 September 2026).

Quantity	Value
Films made; in gallery; quarantined	27; 26; 1 (convergence 0.106)
With sound; with captions	6; 4
Corrected ledger entries	1 (six-frame film moved out of the gallery with a written reason)
Render cost, 264 frames at 512 by 384, service competing	2,232 seconds

Specifications and seeds for every film are in the evidence bundle. The organ satisfies a compatibilist decomposition of agency that the design adopted: it refuses (quarantine and integrity gate), evaluates by its own measure (convergence), acts on its own clock (rounds and stimuli), consults its history (ledger and gallery), and holds ends (families not yet made). What it lacks for a feature is a dramatic kernel, declared as milestones in the charter and not yet admitted.

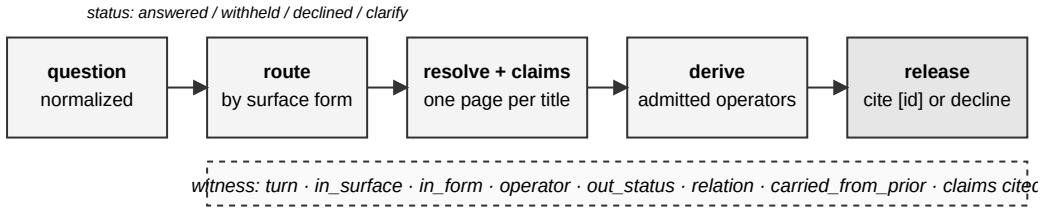

Figure 3. From ledger to release. Every answer on an evaluated route either cites claim identifiers or declines; the witness records the derivation.

9. The release line, 14 to 20 October 2026

Version 2 recorded a deployment gap: the public head imported a frozen letter and served none of the loop’s folds. The release closed it by changing what is public. The released model is the manifest head itself, packed with every file the head opens. The file set was found by tracing the head with an audit hook while it answered the gate’s panel, not by listing directories, after a first tracer missed reads made through a different file interface. Question memory the runtime writes during use is excluded, and a release audit fails the build if any is present. Two variants ship: *full*, with the deep sentence stores and 71 open textbooks (most licensed CC BY-NC-SA, which makes the full variant non-commercial), and *slim*, which replaces those stores with empty stand-ins of the same schema.

Every release passes five steps in order. (a) Each new fold is admitted by the gate of Section 3.5. (b) The packed release is run with the build machine's source tree hidden from it, so that it cannot read anything it does not ship. (c) A sealed test is run once per variant. Its questions are written and hashed before the code freezes, the file is made immutable, and a failure stays failed. The bars are at least 0.90 precision on answered questions, false yeses (answers to questions that must be declined) within a fixed bar, and no format defects. (d) A Merkle root is computed over one SHA-256 per shipped part (each specialist, the gate tools, the fold chain, the Workbench, the launcher and the store manifest), and the author signs a memo carrying the roots on Solana. (e) After upload the release is downloaded again and verified part by part. Step (e) was added by experience: the 14 October release was missing two stores from its upload, found only when a fresh install on a separate machine answered differently, and fixed by patch 1.

Table 7. Release line and sealed tests, each run once per variant. Right / wrong / declined, then false yeses (FY). Format defects were zero unless stated.

Release	Head	Test	Full	Slim	Outcome
Release, 14 Oct	w540	Sealed 19	117 / 3 / 30, FY 1	113 / 1 / 36, FY 0	Released; anchored 15 Oct
Patch 1, 16 Oct	w540	—	Two stores missing from the upload, found by a fresh install; data only		Released
Patch 2, 16 Oct	w543	Sealed 20	129 / 2 / 19, FY 1	128 / 1 / 21, FY 1	Released
Patch 3, 17 Oct	w545	Sealed 21	131 / 2 / 17, FY 0	130 / 2 / 18, FY 0	Released
3.1, 17 Oct	w548	Sealed 22	124 / 3 / 23, FY 1	125 / 2 / 23, FY 0	Released
3.2, 18 Oct	w555	Sealed 23	142 / 7 / 13, FY 3: failed	142 / 3 / 17, FY 1	Slim released; full held
3.2 full, 19 Oct	w556	Sealed 24	119 / 2 / 39, FY 2; 1 format defect: failed	113 / 2 / 47, FY 0; 1 format defect	Not released alone
4.0, 20 Oct	w558	П4-B (186 turns)	157 / 3 / 26, FY 3	156 / 0 / 30, FY 0	Both released

The two failures are kept. Sealed test 23 found three false yeses in the full variant against a bar of two. One was the constitutional minimum age given for "the president of Antarctica". Another was an energy computed from a negative duration with the sign dropped. The third was a walking time read off an unrelated sentence. Each became a class of guard (w556): no stored fact is returned for a question that gives no numbers; a negative amount of something that cannot be negative is withheld; arithmetic is never read from retrieved text. A fourth guard, requiring evidence to name what the question names, lost right answers in every form measured (62, then 21, then 7 on the capability panel) and was not included. The two sealed-23 errors it was aimed at stay open. Sealed test 24 then failed the full variant again on a format bar: a speed asked in km/h was answered in m/h. The full release stayed held until 4.0, which carries 3.2 and passed its own panel.

10. Specialists, the network, and proved kernels

A *specialist* is a small module for one domain (orbital periods, radioactive decay, a star's luminosity) or one family of relations (who directed a film, where a river rises). It is admitted only on a held-out exam that it sees once, and an exam it fails is recorded as failed. Each release ships 310 specialist directories, 161 of them relation readers, beside the gate tools that admitted them. Specialist identifiers fit in one byte, so a plan for a question is a short path through specialists. The path memory holds a whole path as one vector, $\mathrm{PATH} = \Sigma_p \, \mathrm{bind}(K[c_p], R[p])$, in unit

complex phasors of dimension 1,024 drawn from a fixed seed. It recovers four-hop paths exactly. It only orders the plans the network tries and can never make an answer wrong. The network's composition operations (version 6, fold w550) are learned from checked computations, not hand-written. They join specialists into chains, and each chain is admitted on its own held-out cases.

Two rules govern when a specialist may overrule the general route. *Owner-first*: when the specialist a question belongs to was never asked, its whole family is asked, and an answer is kept only if every number the question gives moves it. This test refuses an astronomy reader's orbital speed of Earth (29.8 km/s) for a satellite 400 km above Earth. *Precision before reach*: owner-first never overrules a question that is itself the problem, such as a planet that does not exist or a negative duration.

Proved kernels. For 128 specialists, kernels proved in Isabelle/HOL (Nipkow, Paulson, and Wenzel, 2002), written by ember arlynx and released under CC0, are compiled with MLton (Weeks, 2006) to static binaries. They ship with their checksums. When a specialist answers and its kernel gives the same answer, the reply is marked *proved* and names the theorems. When the kernel gives a different answer, the reply is withheld. Elsewhere nothing changes. What is proved is the computation from the question as read. The reading of the question is not proved, so a proved mark never vouches for the reading. On the specialists' own held-out exams the kernels and specialists agreed on every answer both gave, apart from display rounding.

11. Experiment IV: the law (Oracle7 4.0)

The invocation. Version 4.0 began as a text the author wrote before any design: an invocation of a few hundred words. Its build brief treats it as the authority and makes itself subordinate: where the brief and the invocation disagree, the brief is wrong. The brief reads six principles from it, each citing its words and naming the mechanism of the 3.2 runtime it binds to. The reading is the brief's; the invocation is not reduced to it.

> P1, non-identity with output ("never has become"): the system is never its own answer, so an answer does not become state.
> P2, relation at the smallest unit ("in cantory, spelled out letter by letter"): the twenty-two paths between the Tree's ten nodes are an alphabet, and every route is a word in it.
> P3, throughput rises, state does not ("go faster and faster, but slower in states"): time may fall; the count of steps may not rise to buy it.
> P4, counted steps ("the steeps and the steps"): every ascent is a counted number of steps, logged before release.
> P5, formation by residue ("finally formed into its own known states"): what formed can be re-entered; what did not is kept, typed by where it stopped.
> P6, the law may be wrong ("can you keep up? I'm not sure"): every failure condition must be able to fire.

The Tree in the runtime. Since version 2 the runtime has carried a three-tree walk (question, knowledge and operator trees) on the Tree-of-Life graph. Its ten nodes each apply one operation: READ at Malkuth, FRAME at Yesod, LINK at Chokmah, RELATE at Binah, EXPAND at Chesed, CHECK at Geburah, BIND at Tiferet, COMPARE at Netzach, COMPUTE at Hod, VERIFY at Keter. Its twenty-two edges are the only legal moves. In the tradition the graph comes from, the thirty-two paths of wisdom are the ten nodes and the twenty-two letters, and the letters are the edges (Kaplan, 1997). Version 4.0 takes that literally and makes the edges the typed alphabet Λ.

Implementation. The law is two additive folds above the 3.2 head. Every earlier fold is byte-identical.

The first fold, w557, does four things.

- **Spelling.** The operators a turn logs are mapped to their nodes, and consecutive nodes are joined by shortest paths over the twenty-two edges, ties broken by edge order. The runtime passes through some nodes without operating there, so a turn's word can be longer than its list of operations. "Which country does the Onsen tamago come from?" applies eight operations and spells nine letters, passing through Tiferet and Chokmah. Routes that log no operators are spelled from a route-to-word table frozen before the build. On the release

panel, 23 of the full variant's 186 turns were spelled from logged operators and the rest from the table. The word's length k and the turn's wall-clock time L are fixed before the reply leaves.

- **The ledger.** Every turn appends an entry to an append-only ledger in which each entry carries the hash of the one before. The entry holds the word, k, L and a hash of the question; the question's text is stored only if the operator turns that on. A turn that does not answer adds ρ: the node where formation stopped, the prefix reached, and the reason.
- **Non-identity.** The session's stores are snapshotted before each turn. If the reply answered but stands on nothing (no citation, no proof, not a checked route), they are restored, and the reply is kept only as a discourse record.
- **Recall.** Walks already made are recalled instead of walked again.

The second fold, w558, adds pronoun re-entry. When the chain cannot resolve a short follow-up that names nothing of its own, and the previous answer stood on something and named one person or thing, that answer replaces the pronoun. The question is asked once more, and the reply says so ("Taking 'its' as Japan, from my previous answer.").

Panels and failure conditions. Two panels of 186 turns each were written and hashed before any 4.0 fold was frozen: П4-A (the law panel) and П4-B (the release panel). Each holds 28 follow-up pairs, 30 exact questions, 30 multi-step questions, 20 recurring pairs with the same form and new values, and 30 must-decline questions. Each runs once per arm. The 3.2 arm is the 3.2 head with the same instrument attached and no law, so both arms pay the same logging cost. Four failure conditions were declared in advance:

- **F1, states bought with speed:** faster settings use more steps.
- **F2, nothing formed:** four induction passes over the ledger admit no rule.
- **F3, the law cannot run on 3.2:** building it requires changing a frozen part.
- **F4, no effect:** 4.0 is equivalent to 3.2.

Table 8. Panel results, each arm run once. Re-entrance is the share of recurring forms whose path vectors re-enter at cosine ≥ 0.95, against the same pairs shuffled.

	П4-A, 3.2 arm	П4-A, 4.0	П4-B, 3.2 arm	П4-B, 4.0 full	П4-B, 4.0 slim
Right / wrong / declined	131 / 5 / 50	131 / 5 / 50	157 / 3 / 26	157 / 3 / 26	156 / 0 / 30
False yeses (of 30)	4	4	3	3	0
Follow-up second turns right / wrong	—	—	23 / 0	23 / 0	21 / 0
Mean k (steps)	4.48	4.48	4.95	4.95	4.52
Median L (ms)	63.1	54.6	119.3	120.4	112.0
Ledger entries with ρ	76	76	53	53	60
Re-entrance (shuffled)	93.6% (10.9%)	93.6% (10.9%)	94.0% (16.5%)	94.0% (16.5%)	94.4% (20.6%)
Answers kept out of state	—	4	0	3	8

Throughput and state. The dose-response run took 162 questions at three settings: recall off, the first pass with recall on, and recalled. Mean step count was 5.64 at every setting, and the route word was the same for all 162 questions in all three. Mean time fell from 637 to 589 to 530 ms. The slope of k on L is zero, so F1 does not fire, and speed and step count are shown to be separate measures on this topology.

Findings, including the declared failures.

- On Π4-A, 4.0 and the 3.2 arm gave identical answers on every item. The law changed the record of each answer, its spelling, count, time and ledger, but no answer. On the item comparison this is F4. The pre-registered paired equivalence tests on k and L are not reported in this version; the item-level files are published.
- The four induction passes each drew 150 development questions and handed the ledger's residue to the existing loop. They proposed no rule: no cluster of three or more entries shared a stopping node, reason and form. F2 fires, and the prediction that a typed ledger would yield more admitted rules than the untyped failure inbox is not supported.
- F3 did not fire. The law ran on 3.2 without changing a frozen part. One planned structure moved: the ledger tools ship inside the fold folder rather than as a separate part.
- Pronoun re-entry fired on none of Π4-B's follow-ups, because the 3.2 runtime already resolves them. Its gain is shown only on follow-ups such as "What is its capital?", where 3.2 withheld the answer.
- The panel generator gave the two turns of each Π4-A follow-up pair different sessions, so every Π4-A follow-up was uninformative in both arms. The runner was corrected before Π4-B ran; the sealed panel files are unchanged.

The release bars were met: precision on answered questions of at least 0.90, and false yeses no more than the 3.2 arm on the same panel. 4.0 was released in both variants on that basis. Whether the law confirms the invocation is a separate question, and on this evidence the answer is partial. P1 to P4 hold as implemented and measured. P5's re-entrance is high against its baseline, but the form it predicts would follow, rules from residue, has not appeared.

12. Evaluation, limits, and what the evidence supports

12.1 Limits

Most of the 281 thousand titles are held at stub depth. Four of ten probes on the day's set still fail on the staging head. No rendered essay has passed blind scoring; the essay route now opens on synthesis leads with citations but takes three sentences per page in ordinal order and pads with neighbors chosen by string prefix. Kernel recovery is lexical and a third to a half. Figures are spheres. The held-out sets the gate judges against are authored by humans, so the gate's soundness rests on their design. Version 2's public head served a frozen letter; since 14 October the public release is the manifest head itself (Section 9). On both 4.0 panels the law changed no answer, and no rule has yet been induced from its ledger (Section 11). Coverage remains the weakness: on the release panel the full variant declined 26 of 186 turns. Two errors from sealed test 23 remain open.

12.2 Statistical framing

Zero false affirmatives among 177 observed affirmatives does not establish a true rate of zero; the rule-of-three 95% upper bound is about 1.7%. Six direct answers of ten carries the Wilson interval 0.31 to 0.83. The relational engine's MRR figures are reported with their dataset, baseline, and transductive limit in Section 3.4 but without variance across seeds, which was not recorded. Continuity is reported with denominators in Table 3. Three admissions in fifteen minutes is throughput. The convergence threshold is numerical agreement under one probe.

For version 3: no wrong answer to an answerable question among 157 on the release panel bounds the true rate at about 1.9% (rule of three); three false yeses in thirty carry the Wilson interval 0.035 to 0.256, and none in thirty for the slim variant an upper bound of 0.10. Each sealed test and panel is one run on one set of items. The dose-response latencies are single runs per setting, without variance.

12.3 Claims, narrowed to the evidence

Claim-addressed release. For the evaluated routes, the runtime enforces release of claim-addressed material or explicit decline; unsupported generation is excluded at that boundary by construction. This is not a universal

guarantee over every derivation, relation selection, title resolution, or realization, and Section 5 records a title resolution that failed.

Relocated forgetting. Append-only storage prevents destructive parameter interference. It does not eliminate behavioral regression: routing precedence, alias collision, resource limits, method shadowing, and incomplete gate coverage can make old competence inaccessible, and each has occurred. The design relocates forgetting risk from distributed interference to explicit admission, dispatch, conflict resolution, and gate-coverage failures, each inspectable in a ledger.

Frozen artifact versus cumulative process. The comparison is not between the Oracle and fitted-model systems as a class, which can be retrained, fine-tuned, and equipped with tools and memory. It is between a frozen parameter artifact, whose changes must be inferred from outside, and an explicitly cumulative admitted process, whose changes are a sequence of ledgered state transitions under a non-regression policy. The contribution is inspectable cumulative change, not inevitable superiority.

The process statement, with its conditions. A cumulative process passes a fixed artifact on a defined metric only given continued resources, sound admission, nonzero useful progress on that metric, and no asymptotic ceiling below the artifact. Absent those conditions the statement is nearly tautological. We therefore state the rate and the metric and let the reader judge the conditions: on the gate's own dimensions, over eleven days, the chain advanced 99 admissions with a non-regression policy, and the grounding rate rose from 0.28 to 0.65 on a population that grew from 24 to 397.

12.4 Matching a received shape

The most consequential property of the design is not that it matches frontier models, which it does not, but that it is built to receive the *shape* of what a frontier model does and then to match that shape under its own gate. A shape, in this paper's sense, is a typed structure rather than an output: the eleven slots and nine moves of a concept article (Section 6), the discourse moves that a reference dialogue contains and a candidate lacks (the residue protocol of 31 August, which found seven missing moves in a first run), the schema of a season specification, the procedure by which a synthesis was written. A fitted model cannot receive a shape; it can only be shown outputs and adjust its weights toward them. The Oracle receives a shape as data, fills it by route, renders it, measures the residue between its rendering and the reference, and treats the residue as the next curriculum. This assigns the frontier model a specific and bounded role: not to supply answers, which would be copied, but to supply shapes and references, which are admitted, attributed, and then matched at whatever rate the gate admits. Every experiment in this paper is an instance. Experiment I supplied sentences under a shape (genus lead, mechanism, founders, limits); Experiment II wrote the shape down and measured how much of the reference it recovered; the protocols of Section 13 are shapes for capabilities the system does not yet have. The narrowed claim is therefore this: the system matches received shapes at the rate its gate admits, with the residue measured at each step, and the frontier model's contribution is provenance-marked and revocable like any other source.

13. Protocols declared and not yet admitted

Two protocols were declared on 2 September and entered in the charter so that the loop is their author. Open conversation decomposes into three organs and a channel: an interlocutor state per session, readable by every route and writable to the corpus by none; a selection value over admitted claims that rewards novelty and answered open questions; a question emitter that fills kernel slots by route and turns an empty required slot into a typed request; and a curriculum channel that fills requests under the admission law. Success is defined in advance as counts on thirty unscripted five-turn dialogues, blind-scored. The film protocol reuses the first two organs and adds a dramatic kernel sourced from the world ledger of fifty-six agents, a studio bridge to a procedural-animation renderer with a local voice model on a second machine, math interludes from the render organ, an assembler with ducked score, and dialogue derived only from admitted claims and ledger events. None of these is admitted. They

are release narratives for the future, not completed technologies, and the question emitter in particular is the criterion for the next transition: the point at which the system's intake ceases to depend on its authors.

14. Role in a release program

The paper sits above a portfolio of separable technical releases and does not prove them. Each release's relation to this paper is one of five kinds: direct evidence (tested in a reported experiment or table); operational description (implemented and counted, not isolated experimentally); architectural implication (follows from the design, no direct evidence); declared protocol (specified, not admitted); or outside scope. On that classification the releases with direct or operational standing are the provenance corpus compiler and claim-addressed knowledge ledger (Experiment I, Table 1); title resolution and the release arbiter (Section 3.1, Section 5); the typed knowledge kernel (Experiment II); the additive successor runtime, admission gate, and recursive fold production line (Section 3, Table 2); the mutation-before-improvement tester and the derivation witness (Section 4, Figure 3); and the deterministic media integrity gate (Experiment III). The holographic operator substrate has direct evidence in a construction-time experiment whose full record is not yet published (Section 3.4). The open conversation and film protocols are declared and forthcoming.

15. Conclusion

We have described a system in which knowledge, competence, modification, evidence, and refusal are independently addressable objects, so that its growth can be read as a sequence of admitted state transitions rather than inferred from a new snapshot. Version 2 tied every figure to a runtime identity on one day. Version 3 adds a public release line in which every release is frozen by a root the author signed, every release passed a test written before its code froze, and the two that failed are reported as failed. The 4.0 experiment shows the same discipline applied to a law written as an invocation. The law was built without touching a frozen part. Every answer now carries a counted route and every unanswered turn a typed place. It changed no answers, and its ledger has not yet taught the loop a rule; both are reported as results. The design does not match fitted models on coverage or fluency today. What it establishes is narrower and, we argue, more durable: a machine can hold its knowledge as a ledger, its competence as rules it wrote and can read, and its growth as a gate it applies to itself, and a reader can check every one of those from the released repositories.

Methods note on AI assistance

For version 2, the assisting model (Claude, Anthropic Fable 5.1) wrote the fifteen concept syntheses of Experiment I, the three corpus compilers, the kernel schema, kernels, and measurement script of Experiment II, the render organ successors of Experiment III, the one-call status script that produces the operational figures, and the text of that manuscript. For version 3, the assisting model (Claude, Anthropic Opus 5.5) wrote folds w556 to w558, the 4.0 panel generator and runners, the release, packing and verification scripts, and the text of the new sections. All of it was under the author’s direction and review. The loop authored the majority of the admitted folds. The author designed the system, its laws, its protocols and every experiment, wrote the 4.0 invocation, set the release bars, decided every release, and signed every anchor. The author is accountable for every claim.

References

Aho, A. V., Lam, M. S., Sethi, R., and Ullman, J. D. (2006). *Compilers: Principles, Techniques, and Tools*, 2nd ed. Addison-Wesley.

Buneman, P., Khanna, S., and Tan, W.-C. (2001). Why and where: a characterization of data provenance. *Proceedings of ICDT*.

Ceri, S., Gottlob, G., and Tanca, L. (1989). What you always wanted to know about Datalog (and never dared to ask). *IEEE Transactions on Knowledge and Data Engineering*, 1(1), 146–166.

Doyle, J. (1979). A truth maintenance system. *Artificial Intelligence*, 12(3), 231–272.

Garcez, A. d'A., and Lamb, L. C. (2020). Neurosymbolic AI: the 3rd wave. *arXiv:2012.05876*.

Gayler, R. W. (2003). Vector symbolic architectures answer Jackendoff's challenges for cognitive neuroscience. *Proceedings of ICCS/ASCS*.

Green, T. J., Karvounarakis, G., and Tannen, V. (2007). Provenance semirings. *Proceedings of PODS*, 31–40.

Gulwani, S., Polozov, O., and Singh, R. (2017). Program synthesis. *Foundations and Trends in Programming Languages*, 4(1–2), 1–119.

Hoare, C. A. R. (1969). An axiomatic basis for computer programming. *Communications of the ACM*, 12(10), 576–580.

Kaplan, A. (1997). *Sefer Yetzirah: The Book of Creation*, rev. ed. Weiser Books.

Kanerva, P. (2009). Hyperdimensional computing. *Cognitive Computation*, 1(2), 139–159.

Kleyko, D., Rachkovskij, D. A., Osipov, E., and Rahimi, A. (2022). A survey on hyperdimensional computing aka vector symbolic architectures. *ACM Computing Surveys*, 55(6).

Le Goues, C., Pradel, M., and Roychoudhury, A. (2019). Automated program repair. *Communications of the ACM*, 62(12), 56–65.

Leroy, X. (2009). Formal verification of a realistic compiler. *Communications of the ACM*, 52(7), 107–115.

Lewis, P., et al. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. *Advances in Neural Information Processing Systems*, 33.

Liu, X., et al. (2023). AgentBench: evaluating LLMs as agents. *arXiv:2308.03688*.

Lopez-Paz, D., and Ranzato, M. (2017). Gradient episodic memory for continual learning. *Advances in Neural Information Processing Systems*, 30.

McCloskey, M., and Cohen, N. J. (1989). Catastrophic interference in connectionist networks. *Psychology of Learning and Motivation*, 24, 109–165.

Merkle, R. C. (1988). A digital signature based on a conventional encryption function. *Advances in Cryptology, CRYPTO '87*, LNCS 293, 369–378.

Muggleton, S. (1991). Inductive logic programming. *New Generation Computing*, 8(4), 295–318.

Necula, G. C. (1997). Proof-carrying code. *Proceedings of POPL*, 106–119.

Newell, A., and Simon, H. A. (1976). Computer science as empirical inquiry: symbols and search. *Communications of the ACM*, 19(3), 113–126.

Nipkow, T., Paulson, L. C., and Wenzel, M. (2002). *Isabelle/HOL: A Proof Assistant for Higher-Order Logic*. LNCS 2283. Springer.

Plate, T. A. (1995). Holographic reduced representations. *IEEE Transactions on Neural Networks*, 6(3), 623–641.

Shannon, C. E. (1948). A mathematical theory of communication. *Bell System Technical Journal*, 27, 379–423, 623–656.

Weeks, S. (2006). Whole-program compilation in MLton. *Proceedings of the ACM SIGPLAN Workshop on ML*, 1.

Appendix A. Evidence bundle and reproducibility

Published beside this paper at prometheus7.com/papers as `evidence_bundle_2026_09_02.json`: the measurement timestamps; sha256 and byte size of each of the five store indexes; claim-id prefix counts for the two sentence stores; the gate ledger's sha256 and its reconciliation (evaluation events, unique candidates, repeated evaluations, final dispositions); the candidate-directory disposition counts; the admitted-directory census; the first and last ledgered baselines with denominators; the manifest tip and its sha256; the public and staging harness entry points and their start times; the frozen evidence runtime's manifest hashes; and the charter's sha256. Also published: the film specifications with seeds for the gallery, and the kernel schema, kernels, and measurement report. Withheld to preserve the gate: the held-out turn corpus itself, whose hash is published. Not yet published, and stated as such: a versioned source archive with an environment lockfile (the source tree is not a git checkout; an archive hash will accompany the next version), and the full record of the construction-time relational engine. Until those are published, the appendix establishes internal auditability rather than external reproducibility.

Appendix B. Release evidence (version 3)

Each release repository (huggingface.co/psiloceyeben/oracle7 and oracle7-slim) carries `RELEASE_HASHES.json`: one SHA-256 per part, the Merkle root, the recipe (the same code ships as `workbench/verify.py`, so a user recomputes the root on their own install), the anchor transaction, and the root it supersedes. The mirror at github.com/psiloceyeben/oracle7 (commit `87b015e` for 4.0) carries the fold chain under `runtime/admitted`, both release hash files, and in `research/sealed_history` the sealed panels and their results: sealed tests through 24, Π4-A and Π4-B (hashes `5545c566`… and `ead1f549`…) with their generator, every arm's result file, and the dose-response result. The 4.0 runners are in `evaluation/v4`. The invocation and the build brief are held by the author; the six principles are quoted from it in Section 11. Withheld, as in version 2: the gate's held-out turn corpus, whose hash is published.