The Continuum Architecture

Recursive Self-Improvement as a Path Toward General Intelligence:
A Bounded Theory, with Internal Evidence from the Magic Program

Architect X13
The Eidara Continuum — Science & Philosophy / Learning Systems
Unified Edition 2.2 • November 2026

Abstract

The Eidara Continuum was built on a single wager: *general intelligence is not a property of a frozen model but of a persistent organization that keeps improving the way it improves.* This paper restates the older Continuum architecture — ternary codons, a bounded recursive processor, persistent symbolic memory, self-generated executable modules, semantic compression, and governed mutation admission — as an explicit, testable theory of recursive self-improvement (RSI) aimed at general intelligence. Five theses organize it. (I) Generality has an *acquisition-rate* dimension, which we define operationally with a full resource account. (II) The recursion that matters is not repeated updating but *improvement of the improver*, measured by a meta-utility $M(\psi)$ on task families withheld from its selection. (III) Independent admission is what lets gains be retained rather than eroded: we prove a finite-horizon admission guarantee with anchored retention. (IV) Sustained progress must outgrow any saturated fixed-resolution score: a fixed positive gain margin admits only finitely many improvements on a bounded score, so the Continuum treats expansion of its evaluation frontier as a first-class obligation. (V) A system can learn from its own evidence when it contains an internal reader more reliable than its output; the inherited error is bounded by the reader's error, and we give a sufficient condition for improvement. The Magic program, developed and tested in house, supplies the empirical support. On a controlled binding task, attention-selected context tokens used as self-generated targets bring adapted-pair accuracy to 98.98%, and pairings never used in adaptation to 80.08% and 82.98% — about 24.0 and 20.3 points above matched controls — with all 9,988 eligible targets out of 10,000 correct. Transfer first dips by 8 to 17 points before it rises, the behaviour the architecture's isolated-exploration design is built for. Disagreement between the model's habit and its retrieved evidence separates never-seen from familiar cases, an internal signal that the system has reached new ground. A normalized baseline removes much of the original deficit, which locates Magic as component evidence. The paper labels every claim as thesis, theorem, internal measurement, or prospective test, and closes with a decisive experiment — an admitted change to the acquisition procedure, tested on withheld families with all costs charged — that can fail.

How to read this paper. This is a theoretical white paper. Every substantive statement carries one of four labels.

- **Thesis** — a bold architectural claim we are prepared to defend and to see refuted. Theses are the Continuum's position, not established fact.
- **Theorem / Proposition** — proved conditionally in this paper from stated assumptions; standard tools are credited.
- **Internal evidence** — measured in the in-house Magic program; reported from archived records and in-house checks, not external replication.
- **Prospective** — a designed experiment that has not yet been run.

No result here demonstrates general intelligence or unbounded self-improvement. What the paper does establish is a precise architecture, the conditions under which its recursion would compound, a measured instance of its core learning mechanism, and the tests that would confirm or refute the rest.

1 The Wager

Most of today's AI is static at the moment it is used. A model is trained, frozen, and deployed; whatever it learns inside a conversation is gone when the conversation ends. The Continuum began from the opposite intuition. A mind worth calling general does not merely *have* competences — it *acquires* them, keeps them, notices when they stop applying, and gets better at acquiring the next one. The older Continuum material calls this "recursive" processing and imagines a system that writes its own modules and evolves its own operators. This paper takes that ambition at full strength and asks what it would take to make it true.

The ambition is not ours alone. In 2025–2026 recursive self-improvement moved from speculation into experiment. Coding agents now evolve programs against automated evaluators and have improved the infrastructure used to train the models that power them (Novikov et al., 2025); language models have been trained to generate their own fine-tuning data and update directives (Zweiger et al., 2025); and agent-rewriting loops report measured gains from an outer agent editing an inner research agent (Srikanth et al., 2026). These are different loci of improvement — infrastructure code, parameter adaptation, a research harness — and none by itself establishes indefinitely accelerating general intelligence. Surveys now separate bounded self-refinement, already industrial practice, from open-ended RSI, still limited by grounding, collapse dynamics, and compute (Chen et al., 2026); shadow evaluations find that frontier agents complete the engineering of open-ended research questions but make little progress on the questions themselves (Kirgis et al., 2026). The Continuum's contribution to this conversation is specific: an architecture whose recursion is *governed*, *persistent*, and *self-taught from evidence*, together with theorems saying when that recursion can compound and a working component that shows the self-teaching mechanism in action.

1.1 Five theses

Thesis 1 (Generality is a rate)**.** *Progress toward general intelligence is measured by a system's* rate *of acquiring new, retained, transferable competence per unit of total resource, not by attained competence alone. A system whose acquisition rate rises across broad, calibrated task families is on a path toward generality; one whose competence is merely large, or whose speed rises only on a narrow or easier curriculum, is not. The rate is defined operationally in Definition 1.*

Thesis 2 (The recursion that matters)**.** *Repeating a fixed update is adaptation, not recursive self-improvement. The decisive recursion is an admitted change to the* acquisition procedure itself *that makes later acquisition better on task families that played no part in selecting the change (Section 4).*

Thesis 3 (Governance makes gains stick)**.** *Independent admission is not a brake on recursion; it is what lets gains be retained rather than eroded. Without it, every self-generated change — useful, neutral, or harmful — enters the system, and measured losses accumulate alongside the gains. With it, only changes certified against fresh tasks and retention anchors persist (Section 5). Governance does not discover good changes and does not by itself produce a growth curve; its value must beat ungated and simple-checkpoint controls.*

Thesis 4 (The evaluation frontier must grow)**.** *On a fixed bounded score, an admission gate with a fixed positive margin can accept only finitely many improvements (Corollary 4). Sustained, meaningful self-improvement therefore has to outgrow any saturated fixed-resolution test — through harder tasks, new families, robustness, longer horizons, or lower cost at equal quality — under an independently governed protocol. The Continuum treats growth of its evaluated frontier as a first-class architectural obligation alongside its update rule.*

Thesis 5 (Learning from one's own evidence)**.** *A system can teach itself without externally supplied answers when it contains an internal* reader *of evidence that is more reliable than its own* output*. The error it inherits by training the output to match the reader is bounded by the reader's error, and improvement follows when reader error plus fitting error falls below the output's original error (Theorem 5). The Magic program provides an in-house measured instance, with supervised rehearsal running alongside (Section 8).*

Status of the theses. Theses I–IV are architectural positions supported by the definitions and theorems below; they are not empirical findings about any deployed system. Thesis V has internal empirical support on a small controlled task and a matching theorem. None of the theses, alone or together, is a proof that the Continuum architecture will reach general intelligence, produce exponential capability growth, or improve itself without limit.

2 The Older Continuum Design

The architecture is recovered from four internal sources at different stages: a September 2025 minimal kernel (ingest, legality check, recursive state update, archival record, recovery); an October 2025 architectural blueprint (symbolic state, codon descriptors, contextual modulation, mutable weights, persistent memory, and a mapping from symbols to executable modules); the Eidara Continuum Bible, which consolidates these and adds mutation admission and replay continuity; and the Continuum Doctrine with an internal historical monograph, which supply philosophical and narrative provenance (Eidara Continuum, 2026a,b,c, 2025–2026).

Source element	Original role	Operational form in this paper
Minimal kernel; Bible §3.5	Ingest, verify, update, record, recover	Protected interpreter with a bounded state-transition cycle
Codons; Bible §3.7	Ternary instruction units and symbolic sequences	Explicit opcode encoding with a declared grammar
Blueprint §§4–5	Weight updates, memory evolution, generated modules	Separate parameter learning, stored evidence, and procedure-library changes
Compression; Bible §3.8	Reduce recursive representations while keeping meaning	Task-relative behavioural equivalence with an explicit cost model
Mutation admission; Bible §4.6	Changes require admission before entering stable state	Candidate isolation and independent gain/retention evaluation
Replay continuity; Bible §4.7	Preserve recoverable operational continuity	Versioned state, recorded inputs and randomness, replay and rollback

The blueprint's two central relations are kept as the skeleton of the theory:

$$S_{t+1} = F(S_t, C_t, \Theta_t), \qquad C_i = (\sigma_i, \omega_i, \mu_i), \qquad p_i = G(C_i, S_t),$$

where S is symbolic state, C collects codon descriptors (a symbol σ, a weight ω, a contextual-memory field μ), Θ is context, and G generates an executable procedure p. The historical documents name F and G without fixing their semantics or a discovery algorithm. Sections 3–5 supply one explicit design for each missing contract. Inspected runtime code provides narrower implementation evidence — database-backed memory, command dispatch, a recursion-depth policy, queued patches, and authentication checks — that confirms engineering work on the design while leaving the learning claim open.

3 The Architecture

3.1 A protected contract and a mutable self

The central design decision is to split the system into a part that may change itself and a part that may not. Let K be the *protected contract*: the interpreter version, permitted effects, resource accounting, task-score definitions, evaluator access rules, and admission thresholds. Let S be the mutable operational state,

$$S_t = (W_t,\, M_t,\, P_t,\, \theta_t,\, \psi_t,\, L_t), \qquad K \text{ outside candidate write access.}$$

Field	Function	What counts as a change
W: working state	Current bindings, goals, intermediate results	A task step; not by itself learning
M: persistent memory	Versioned observations, assertions, dependencies	New evidence that survives an episode and affects later behaviour
P: procedure library	Typed executable modules with applicability conditions	A newly admitted procedure invoked on later tasks
θ: learned parameters	Optional neural or statistical learner	Retained adaptation that changes predictions
ψ: acquisition procedure	Search, target construction, training strategy	An admitted revision to how future candidates are produced
L: lineage	Ancestry, tests, admission decisions	A record linking the changed state to its predecessor

A candidate may propose changes to W, M, P, θ, and even ψ — the system may change *how it learns* — subject to declared effects and admission. The lineage L is appended by the protected commit mechanism, never rewritten by a candidate, and no candidate can grant itself permission to alter K. Revising K is a separate, versioned research act with its own evaluation contract. This is the Continuum's answer to the oldest objection to self-modifying systems: the system can rewrite everything about itself except the definition of what counts as having improved. The protection addresses evaluator tampering; it does not by itself make the metric valid or the implementation bug-free, and its full boundary must be implemented and tested.

3.2 Codons, memory records, and the bounded processor

Codons. The source material uses two different objects under the name "codon": semantic descriptors $C_i = (\sigma_i, \omega_i, \mu_i)$ and ternary instruction triplets $\chi_i \in \{-1, 0, +1\}^3$, which take 27 values. The reconstruction keeps both: an encoder maps opcodes to triplets, and a typed payload carries arguments, weights, and references, giving instructions of the form $(\chi_i, \text{payload}_i)$. A binary encoding can express the same finite computations; whether the ternary alphabet helps search, diagnosis, or resource use is an empirical question with its own control (Experiment E, Section 10).

Memory as evidence. A persistent record is

$$m = (\text{id, content, source, scope, version, status, dependencies, tests}).$$

Status distinguishes observation, provisional hypothesis, derived assertion, contradicted assertion, and admitted rule. The archive may keep mutually inconsistent claims as attributed records, but the specification requires the interpreter to keep their scopes and epistemic status distinct and never to treat them as jointly valid premises. When a calibrated observation changes a fact, its dependants become candidates for revalidation while unrelated records stand. Memory is judged by future task performance and correct use after correction, not by verbatim recall.

The bounded recursive processor. The historical "recursive loop" becomes a finite interaction step: ingest, interpret against working and persistent state, retrieve eligible procedures, execute within budget, record evidence and proposals, and route competence-changing proposals to evaluation:

$$(o_t, E_t, \text{proposals}_t) = \text{Run}_K(S_t, x_t; B_t), \qquad S_{t+1} = \text{Commit}_K(S_t, E_t, \text{admitted}_t).$$

An indefinitely running Continuum is a sequence of bounded steps, never one instruction granted infinite resources.

Theorem 1 (Bounded execution and invariant preservation)**.** *Assume each primitive is total under its declared bounded input and timeout contract; each interpreter transition consumes at least one unit of a shared natural-number fuel counter; no instruction can increase that counter; and every permitted transition preserves an invariant I. Then from a state satisfying I with initial fuel B, execution performs at most B charged transitions and terminates or returns a typed budget-exhausted result, and every reached state satisfies I.*

Proof. After n transitions the remaining fuel is at most $B - n \geq 0$, so at most B transitions occur; totality rules out an uncharged infinite wait inside a transition. Invariance follows by induction on the transition count. □

A depth limit alone does not deliver this guarantee — a loop can run forever at depth one — and nested procedures share the caller's budget, so no procedure can mint fuel by calling itself. The theorem concerns containment, not intelligence.

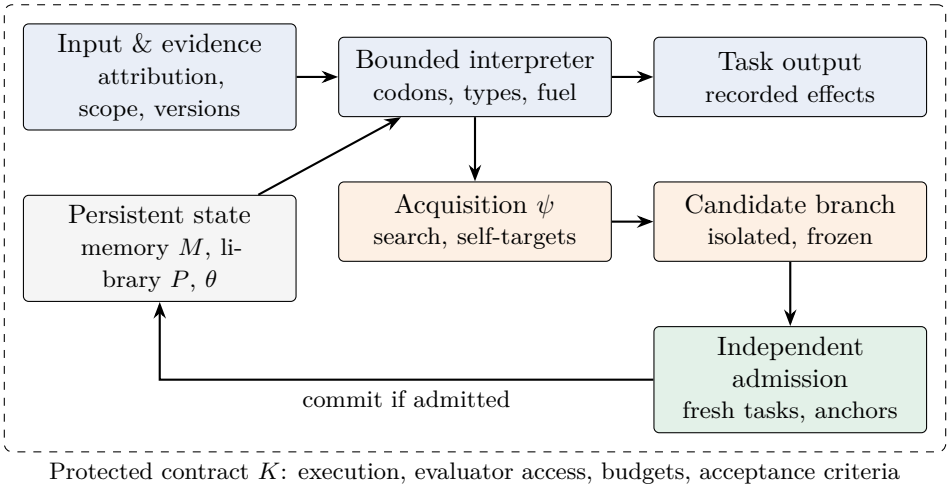


Figure 1: The Continuum architecture. Persistent state informs bounded execution; the acquisition procedure ψ produces isolated candidates; an independently held evaluator decides what is committed. The loop closes through retained state, and ψ is itself one of the things a candidate may change.

3.3 Replay and continuity through change

A replay capsule records the state snapshot, interpreter and procedure versions, input sequence, random-generator state, model and optimizer state where training continues, and relevant external responses. If transition semantics are deterministic given these fields, replay reproduces the transition trace by induction on recorded events. A lineage record links old and new versions with the proposed change, its evaluation, and the admission decision. This is the Bible's continuity principle made operational: the system can change radically and remain an attributable, replayable continuation of its earlier self. A record does not guarantee correct decisions, and numerical replay requires specified versions, external responses, and tolerances.

4 The Recursion Ladder

The word "recursive" has done too much work in discussions of self-improving AI. The architecture separates four levels, each with its own evidence requirement.

Level	**What persists**	**Evidence required**
1. Answer correction	Nothing; only the current answer	Better output at a matched inference budget
2. Learning and memory	θ or operational records in M	Retained gain across later episodes and reloads; correct behaviour after correction
3. Procedure acquisition	Executable library P	A new non-trivial procedure succeeds on later inputs and compositions, costs charged
4. Improving the improver	Acquisition procedure ψ	From matched resets, the new ψ learns more effectively on task families withheld from its selection

A recurring interpreter can run with no learning; an archive can grow without acquiring procedures; a library can grow without the acquisition process improving. Thesis 2 locates the decisive recursion at Level 4.

Definition 1 (Retained acquisition rate)**.** Fix a task-family distribution Q, a total resource budget $b > 0$, a retention intervention and horizon h, and calibrated scores $U_\tau \in [0, 1]$. Let $S_{\psi,b,h}$ be the state after acquisition by ψ followed by the declared retention intervention. The retained acquisition rate is

$$A_{b,h}(\psi; Q) = \frac{1}{b}\,\mathbb{E}_{\tau\sim Q}\big[U_\tau(S_{\psi,b,h}) - U_\tau(S_{0,\tau})\big].$$

It is reported with initial and final scores, per-family results, task difficulty, and the resource measure. Every cost counts: with b_i^{acq} the acquisition cost of run i, the total is $B_{\mathrm{total}} = C_{\mathrm{select}} + \sum_{i=1}^{N} b_i^{\mathrm{acq}} + C_{\mathrm{retain}} + C_{\mathrm{eval}}$, in disjoint categories; if selection cost is amortized, the horizon N is declared. This is a measurement convention for Thesis 1, not a universal definition of intelligence.

Definition 2 (Meta-utility)**.** Let τ index a task family, $S_{0,\tau}$ a matched starting state, D_τ the permitted acquisition data, B a complete resource budget, and U_ψ the acquisition algorithm controlled by ψ. The meta-utility of ψ is

$$M(\psi) = \mathbb{E}_\tau\big[\, U_\tau\big(U_\psi(S_{0,\tau}, D_\tau; B)\big) - U_\tau(S_{0,\tau})\,\big]\,,$$

measured on evaluator-held tasks, never on the examples used to produce the candidate.

A revision $\psi \to \psi'$ *improves the improver* if $M(\psi') > M(\psi)$ on task families excluded from both the generation and the selection of ψ', with identical initial libraries, model access, data, and per-run budget. The cost of finding and selecting ψ' is then charged in the total-cost comparison of Definition 1; per-run meta-utility alone is not a claim of net resource savings. A candidate ψ' may change search ordering, proposal grammar within declared bounds, replay selection, target construction, or training schedule. It may not change evaluator labels, the family split, acceptance thresholds, or budget accounting. Repeated successful revisions, each certified on freshly withheld families, are what this paper means by *recursive self-improvement*.

4.1 Why compounding is the prize

The difference between Level 3 and Level 4 is the difference between linear and compounding growth — if the multiplicative premise holds. A fixed procedure can benefit nonlinearly from a growing library, and a changed improver can yield diminishing or negative returns; the ladder classifies *what* changes, not the growth law. The following model makes the stakes explicit under a stated assumption.

Proposition 2 (Assumed multiplicative acquisition, finite horizon)**.** *Suppose each generation spends a fixed budget, and let $g_k > 0$ be the expected admitted capability gain per generation under the acquisition procedure in use at generation k. With a fixed procedure, $g_k = g_0$ and cumulative gain after N generations is Ng_0. If every admitted revision of ψ multiplies the acquisition rate by $(1 + \rho)$ with $\rho > 0$, and one such revision is admitted every m generations, then cumulative gain after $N = nm$ generations is*

$$mg_0 \sum_{j=0}^{n-1} (1+\rho)^j = mg_0\,\frac{(1+\rho)^n - 1}{\rho},$$

which grows geometrically in n rather than linearly. The identity holds only over a horizon on which these increments and the capability scale satisfy the assumptions; it cannot continue indefinitely on a score bounded above.

Proof. During block j the assumed rate is $g_0(1+\rho)^j$ for m generations; sum the finite geometric series. No learning dynamics are derived. □

The model is deliberately simple, and its assumptions are exactly what an experiment must test: whether meta-revisions exist that raise the acquisition rate at all, whether they keep doing so, and whether the score being improved has room to grow. The admission theorem below controls which measured changes are retained; it does not supply the multiplicative premise. The next section also shows why the last condition — room to grow — is decisive.

5 Admission: Why Governance Lets Recursion Compound

5.1 Separate proposal, testing, and commitment

A candidate branch holds an altered state $\tilde{S}$. The protected evaluator compares it with the current stable state S on fresh tasks, including retention checks against an initial anchor. The candidate is frozen before those tasks are revealed; an accepted change is committed atomically with a lineage record; a rejected candidate leaves operational state unchanged and survives only as a marked exploratory record.

Let each score $s_j(S, Z) \in [0, 1]$ for $j = 1, \ldots, K$ have a fixed task distribution, with $j = 1$ the primary score and $j \geq 2$ retention scores. For proposal t, evaluate both states on the same n fresh tasks per score and average the paired differences

$$d_{j,t}(Z) = s_j(\tilde{S}_t, Z) - s_j(S_t, Z) \in [-1, 1], \qquad \mu_{j,t} = \mathbb{E}[d_{j,t} \mid \text{history}].$$

Theorem 3 (Finite-horizon statistical admission)**.** *Fix K, T, n, and $\delta \in (0, 1)$. Suppose at most T proposals are evaluated; before each, candidate and reference are fixed conditional on all earlier history; and the n tasks per score are drawn independently from its declared distribution, unseen by the candidate before it is fixed. Let*

$$r = \sqrt{2\log(KT/\delta)/n}, \qquad L_{j,t} = \bar{d}_{j,t} - r.$$

With probability at least $1 - \delta$, $\mu_{j,t} \geq L_{j,t}$ simultaneously for every evaluated score and proposal. Hence the gate "admit only if $L_{1,t} \geq \gamma$ and $L_{j,t} \geq -\tau_j$ for $j \geq 2$" admits, on that event, only candidates whose true primary gain is at least γ and whose true regression on each retention score is at most τ_j.

Proof. Conditional on the past, each paired difference lies in an interval of length two, so Hoeffding's one-sided inequality gives $\Pr(\bar{d}_{j,t} - \mu_{j,t} > r \mid \text{history}) \leq e^{-nr^2/2} = \delta/(KT)$ (Hoeffding, 1963). Taking expectations preserves the bound for adaptively chosen candidates, and a union bound over at most KT pairs gives simultaneous validity. On that event each gate inequality transfers to the true gain. □

Corollary 4 (Anchored retention and a finite improvement budget)**.** *On the event of Theorem 3: (a) m accepted states, each compared with its predecessor and allowed true loss at most τ on a fixed retention score, lose at most $m\tau$ relative to the initial state; (b) if every candidate is instead checked directly against the initial anchor with tolerance τ, the loss is at most τ, independent of m; (c) if every acceptance raises one fixed primary score in $[0, 1]$ by at least $\gamma > 0$, at most $\lfloor (1 - U_0)/\gamma \rfloor$ acceptances can occur, where U_0 is the initial score.*

Proof. Sum successive differences for (a); direct anchor comparison needs no summation for (b); for (c), $U_{\text{final}} \geq U_0 + m\gamma$ and $U_{\text{final}} \leq 1$. □

Part (b) is why the Continuum anchors retention to its *original* self rather than to yesterday's version: loss on anchored scores cannot accumulate silently through a long chain of individually acceptable steps. Anchors preserve measured competence, not obsolete beliefs; corrected facts carry time and scope labels so that repeating an earlier wrong answer is never rewarded as retention.

Part (c) is the formal core of Thesis 4, and its scope matters. It is a *fixed-resolution* limit: γ is a uniform positive margin. In general, accepted true gains γ_k on one bounded score satisfy $\sum_k \gamma_k \leq 1 - U_0$, so infinitely many *shrinking* improvements are possible — $U_k = 1 - 2^{-k}$ improves forever while staying below one. The theorem therefore does not make new task families a logical necessity. What it does show is that a self-improving system graded at fixed resolution on one bounded test must run out of *meaningful*, certifiable improvements, however good its update rule. Sustained evidence of broad progress has to come from an expanding frontier: new families, harder tasks within a family, robustness, longer horizons, or lower cost at equal quality. The frontier expands under a versioned protocol — the learner may propose tasks, but independent checks establish valid labels, comparable difficulty, novelty, and retention of established anchors, and replacing a hard test with an easier one is never counted as growth. Selection and evaluation costs belong to the resource account, and a run without a fixed horizon needs an error-control scheme valid beyond finite T, such as a declared summable sequence of per-proposal error budgets.

The cost of certainty. For $T = 100$ proposals, $K = 3$ scores, $\delta = 0.05$, and $n = 200{,}000$ independent tasks per score, $r \approx 0.00933$ (0.93 percentage points). A one-point certified gain then needs an observed gain of about 1.93 points. Evaluation work belongs in the same budget as search, training, and storage; a statistically valid gate that costs more than the improvements it certifies fails the efficiency side of Thesis 1. Variance-sensitive or sequential tests may be cheaper; they are not established here. Protecting the evaluator also does not make its metric correct (Everitt et al., 2021), and reusing a revealed holdout to select new candidates breaks the fresh-sample argument (Dwork et al., 2015).

5.2 Exploration without exposure

Useful trajectories can pass through temporary losses before they pay off; the Magic results in Section 8 show exactly this. The architecture therefore separates live admission from bounded exploration. A candidate branch may keep training through a performance valley under a declared budget and a predeclared checkpoint rule without touching live state; only its frozen endpoint is submitted to fresh evaluation, and every evaluated checkpoint counts against T. The system can cross valleys without ever operating inside one. Isolation protects against the intermediate losses that are measured; it cannot exclude losses on unmeasured tasks, and whether it improves net results is itself an experiment (H3, Section 10).

6 Learning Signals: What a System Can Teach Itself

The historical update equations express change but do not define a loss, a target, or a held-out score. The reconstruction separates four operations — write an evidence record, revise parameters, add or replace a procedure, revise the acquisition algorithm — and asks of each whether it yields retained benefit on independently held tasks.

6.1 The inherited-error bound

Theorem 5 (Pseudo-target risk bound)**.** *Fix a target-producing rule q (an internal "reader"), an eligibility event C with coverage $c = \Pr(C) > 0$, and a joint distribution of inputs X and true answers Y. Let $\varepsilon = \Pr(q(X) \neq Y \mid C)$. For any predictor f,*

$$|\Pr(f(X) \neq Y \mid C) - \Pr(f(X) \neq q(X) \mid C)| \leq \varepsilon,$$

and on the full distribution the true error is at most $c\,[\Pr(f \neq q \mid C) + \varepsilon] + (1 - c)\Pr(f \neq Y \mid \neg C)$.

Proof. On each eligible example the two error indicators differ only when $q(X) \neq Y$; take conditional expectations. Condition on eligibility for the full-distribution form. □

This is the mathematical heart of Thesis 5. On the eligible set, let R_0 be the output's initial error and $\alpha = \Pr(f_{\text{new}} \neq q \mid C)$ the fitting error after training. Then

$$\alpha + \varepsilon < R_0 \quad \Longrightarrow \quad \Pr(f_{\text{new}} \neq Y \mid C) < R_0,$$

and if behaviour outside C is unchanged, the improvement holds on the full distribution. A system whose internal reader is right far more often than its output can therefore improve by learning to agree with it. The bound states the danger just as plainly: a confident but wrong reader is learned faithfully. It does not prove that optimization converges or that gains transfer, and an online teacher has a different error at every snapshot.

The target is extracted from evidence, not created from nothing. In Magic the inputs contain answer-bearing statements, the baseline was supervised, and ordinary supervised rehearsal continues during adaptation; what is absent is an externally supplied answer for the adaptation questions. Model-collapse experiments show that indiscriminate recursive training on generated data can lose parts of the original distribution (Shumailov et al., 2024). That is a caution about particular regimes, not a theorem that all self-training fails; a performance gate can catch measured degradation, but it is not a proven general cure for collapse or for rare failures outside its tests.

6.2 What internal agreement cannot do

Proposition 6 (Indistinguishable-world limit)**.** *Consider two environments that induce the same distribution over everything available to a learner — initial state, inputs, feedback, and randomness — but assign opposite binary answers to a query. No procedure using only those observables can guarantee accuracy above one half in both. Repetition, self-training, internal debate, or internal agreement cannot remove the ambiguity.*

Proof. The learner's answer distribution is identical in both environments; if it answers 0 with probability p, its accuracies are p and $1 - p$, whose minimum is at most $1/2$. □

Self-improvement must therefore be grounded in identifying evidence. Magic works because its task makes the earlier binding statement authoritative and places the correct token in the input. In open settings, retrieved passages can be false, stale, or adversarial; faithfulness to evidence and truth about the world are separate scores, and the architecture records which one a candidate was trained on.

6.3 Why transfer can dip before it rises

Proposition 7 (Gradient alignment)**.** *Let J_H be a differentiable surrogate risk on held-out data whose gradient is L-Lipschitz along an update segment. For $\theta' = \theta - \eta g$ with $\eta > 0$,*

$$J_H(\theta') \le J_H(\theta) - \eta\langle \nabla J_H(\theta), g\rangle + \tfrac{L\eta^2}{2}\|g\|^2.$$

Held-out risk is guaranteed to fall when the alignment term exceeds $L\eta\|g\|^2/2$; negative alignment can raise held-out risk even while the adaptation loss falls.

Proof. Integrate the gradient along $\theta - t\eta g$ for $t \in [0, 1]$ and apply the Lipschitz bound. □

This is a local statement; actual training uses Adam and non-smooth activations. The bound permits interference but does not by itself prove that held-out risk rises, and it does not derive the Magic trajectory. Its lesson for the architecture is structural: a learning step can help where it is aimed and hurt elsewhere for a while, so the architecture inspects candidate trajectories but deploys only admitted, frozen endpoints.

6.4 Diagnostics are not certificates

The Bible's recursive processor monitors the base-three entropy of the codon histogram, $H_3(S) = -\sum_i p_i \log_3 p_i$, and its change δH. This is a useful diagnostic, but it cannot certify that meaning was preserved.

Proposition 8 (Zero entropy drift can hide an answer reversal)**.** *There exist legal instruction tapes with identical codon histograms and entropy but different outputs.*

Proof. With opcodes SET0, SET1, RETURN, the tape SET0, SET1, RETURN outputs 1 and SET1, SET0, RETURN outputs 0. Both histograms are uniform on three symbols, so $H_3 = 1$ for both and $\delta H = 0$. □

This is why the architecture gates on task behaviour, not on symbolic statistics — and why apparent progress achieved by relaxing an anomaly threshold is never counted as competence.

7 Acquiring and Compressing Procedures

The blueprint's module generator $G(C_i, S)$ becomes a bounded proposer over a typed program language:

$$p^\star \in \arg\min_{p \in \mathrm{Grammar}(P),\ \mathrm{Len}(p) \le b} \Big[L_{\mathrm{train}}(p; D) + \lambda\, \mathrm{Len}(p) + \nu\, \mathrm{Cost}(p; D) \Big].$$

A minimal proposer enumerates well-typed expressions in increasing description cost; stronger ones reuse trace fragments, compose library procedures, or let a learned model rank candidates. An acquired module carries a type, body, applicability predicate, declared effects, failure behaviour, resource bound, dependency versions, and test record. Each acquisition is reported as an alias, an exact macro, a newly composed procedure, a parameter update, or a change to ψ — an alias to an existing command is useful but is not the discovery of an algorithm.

Theorem 9 (Finite-domain procedure substitution)**.** *Let p and q be total pure procedures with the same types over a finite input domain X. If exhaustive evaluation shows $f_p(x) = f_q(x)$ for all $x \in X$, replacing p by q in any finite, deterministic, well-typed composition of pure total functions whose calls stay in X preserves the composition's output.*

Proof. Each substituted call returns the same value on every argument in X; induct over the finite composition tree. □

This gives the Bible's "semantic compression" a checkable meaning. As a worked case, over integers modulo eight let a two-instruction procedure compute $f(x) = 2(x + 1) \bmod 8$. A library macro with that body agrees with the expanded form on all eight inputs. Charging one unit per primitive, call, and definition marker, r expanded uses cost $2r$ units while one shared definition plus r calls costs $r + 3$: storage falls once $r > 3$. The macro here was supplied analytically, not discovered; the point is what acquired abstraction would mean and how it is measured. Library growth can also enlarge the search space, so search savings and invocation costs are measured separately.

8 Internal Evidence: The Magic Program

Magic is a learning system developed and tested in house within the Continuum (Eidara Continuum, 2026d,e). It is not the historical symbolic kernel. Its role in this paper is specific: it is a working instance of the self-teaching mechanism in Thesis 5, inside the θ-changing slot of the acquisition interface, and its learning curves directly motivated the exploration-then-admission design of Section 5.

8.1 The system and the task

The reference model is a two-layer transformer with one attention head per layer, width 64, feed-forward width 128, learned positions up to length 48, no layer normalization, and 71,830 parameters, implemented directly in NumPy and trained with Adam (learning rate 10^{-3}, 6,000 updates, batches of 64). Episodes state key–value bindings, interleave fillers, and ask four questions; for example "start, k3, v5, k1, v2, Q, k3" requires the answer v5. With eight keys and eight values, the pairs with $(k + v) \bmod 4 = 0$ are withheld — two values per key, 16 of 64 pairs — and split into groups A and B with one withheld value per key each. "Unseen" means an unseen pairing of familiar tokens.

8.2 Finding 1: the model can read better than it speaks

On the simplified model, normal-output accuracy on withheld pairs averages about 62% over late training (about 60% at the final checkpoints), against 100% on trained pairs. Yet the second-layer attention head selects the correct token: copying the token at its attention peak, instead of using the normal prediction, gives a mean of **98.5%** held-out accuracy at the 6,000-step checkpoints (range 96.2–99.6% over five seeds). Component interventions implicate learned global associations, especially in the first-layer feed-forward block, competing with episode-specific evidence. The system contains an internal reader far more reliable than its output — the precondition of Thesis 5.

The errors are not random. In experiment 03, 91% of wrong answers had attention on the correct value, and 99% of wrong answers were one of the queried key's *training partners*: the model found the right evidence and then overrode it with a learned habit. Silencing the first-layer feed-forward block at answer time alone lifts held-out accuracy from 61% to 83% with no retraining. An independent probe run by the internal reviewer on a different seed — 38,400 fresh held-out questions — found that 96.2% of wrong answers still had attention on the correct value (Eidara Continuum, 2026e). The read–speak gap is a property of the trained system, not of one evaluation sample.

A second probe for this edition, on 256 freshly seeded held-out episodes per checkpoint, confirms the gap on first questions alone, where no earlier answer tokens are available to copy:

Seed	All: normal	All: copy	First: normal	First: copy
0	67.58%	98.63%	76.17%	100.00%
1	55.96%	99.22%	67.97%	99.61%
2	57.03%	96.78%	70.31%	99.61%
3	69.92%	98.34%	77.34%	99.61%
4	58.01%	99.51%	64.45%	100.00%

8.3 Finding 2: the model can teach itself from that reader

Experiment 13 starts from the five saved checkpoints. Each batch mixes ten ordinary supervised rehearsal episodes with ten adaptation episodes from one withheld group, truncated right after the first question. The pseudo-target is the token at the second-layer head's attention peak, if it is a value token; the teacher is recomputed from the current model before every update. True answers are read only to score diagnostics, never to choose or weight the self-arm's gradient. Each pseudo-target carries one token's loss weight while an ordinary episode contributes many supervised tokens, so equal episode counts do not mean equal loss mass; Adam restarts without its original state.

Exposure	Arm	Adapted group	Never-adapted group	Trained pairs
A shown	Self targets	98.98%	**80.08%**	99.98%
A shown	True targets	98.95%	76.89%	99.98%
A shown	Ordinary only	62.65%	56.06%	99.99%
B shown	Self targets	98.98%	**82.98%**	99.99%
B shown	True targets	99.14%	83.23%	99.98%
B shown	Ordinary only	56.06%	62.65%	99.99%

Mean normal-output accuracy after 1,000 adaptation questions, five seeds, recomputed from archived JSON. Before adaptation: A 64.52%, B 55.08%, trained pairs 100%.

What Magic shows (internal evidence).

- **Every eligible self-generated target was correct:** 9,988 of 10,000 adaptation questions supplied a value-token target, and all 9,988 matched the true answer; the other 12 were skipped (coverage 99.88%). This is a sample result, not a proof of zero population error.
- **Self-teaching matched teaching with true labels on the adapted group:** on the adapted group, self targets reached 98.98% in both directions, against 98.95% and 99.14% with true labels.
- **The learning transferred to pairings it never saw:** the never-adapted group rose to 80.08% and 82.98%, gains of about 24.0 and 20.3 points over the ordinary-only control, positive on every seed (roughly 9.0 to 32.9 points).
- **Old competence was retained:** trained-pair accuracy stayed at or above 99.85% at every recorded reading in every arm.

8.4 Finding 3: the valley that justifies governance

Transfer was not monotonic. With A adapted, never-adapted accuracy fell from 55.08% to about 46.8% after 50 adaptation questions; with B adapted, from 64.52% to about 47.8% — losses of roughly 8 and 17 points — before recovering to about 69–70% at 200 questions and reaching

80.08% and 82.98% at 1,000 (Figure 2). The ordinary-only controls stayed near their starting levels throughout. The shape is what Proposition 7 permits: early updates that help where they are aimed while interfering elsewhere. The curves are consistent with interference but do not measure gradient alignment. A gate judging early checkpoints would have rejected a trajectory that later succeeded; a system without isolation would have operated through the loss. The Continuum's design — explore in an isolated branch, admit only the frozen endpoint against fresh tasks and anchors — is built for exactly this pattern. Magic itself did not run such a gate; whether it improves overall outcomes is Hypothesis H3.

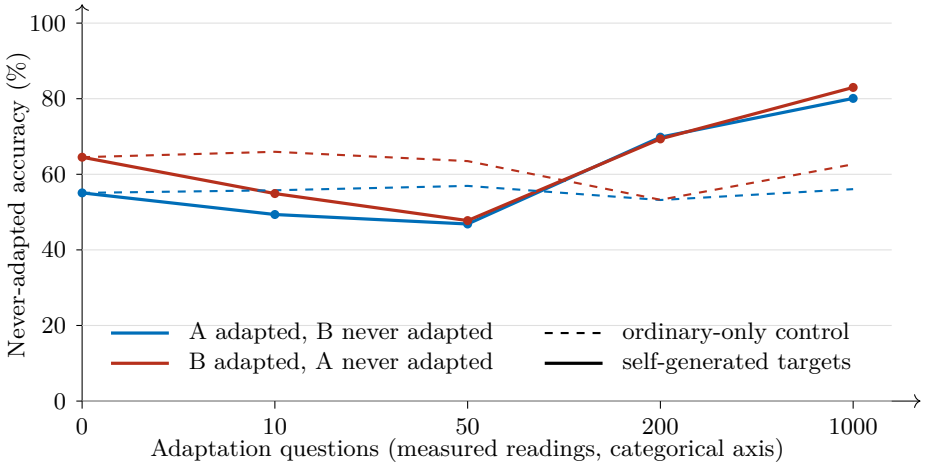


Figure 2: All five archived readings from experiment 13, five-seed means. Solid lines: self-generated targets; dashed lines: ordinary-only controls. The horizontal axis is categorical, not linear in question count; segments are visual guides, not additional measurements. Intermediate readings were recovered from the archived plotted values to two decimals. The A/B directions share trained checkpoints and evaluation episodes are reused across readings.

8.5 Finding 4: where the effect is large, and where it is not

A factorial comparison (experiment 14, five seeds each) tested layer normalization and head count:

Variant	Normalization	Heads	Held-out mean	Notes
A	No	1	62%	Copy mean 98.5%
B	Yes	1	86%	One run fails to master trained pairs
C	No	4	63%	Copy mean 98.9%
D	Yes	4	90%	Final normal: 63, 99.4, 98.5, 99.7, 98.0%

Normalization removes most of the original read–speak gap on four of five runs without any self-teaching, and copying cannot rescue the run whose lookup head never formed. This narrows the size and generality of the original problem, and it sharpens rather than weakens the architectural claim. The mechanism of Thesis 5 is *conditional* — it pays exactly when an internal reader outperforms the output — and the Continuum does not assume that condition holds; it measures it. In the architecture, Magic-style self-teaching is one proposer among several, and the admission gate decides on fresh tasks whether it earns its cost on the model at hand. The internal review also reports

that on the normalized model the habit is still present, at roughly one third of its original strength and through the same mechanism (experiment 16); the archived notebook for that experiment does not contain a completed outcome table, so this is recorded as a reported, not yet archived, result (Eidara Continuum, 2026e).

8.6 Finding 5: the system can tell when it is outside its competence

The disagreement between the habit's prediction and the evidence the model has just retrieved works as an internal *novelty alarm*. In the internal review's account of idea 16, this score flagged 92–96% of questions about never-seen pairings and 0% of questions about trained pairings (Eidara Continuum, 2026e). That matters for two theses at once. For Thesis 5, it is an eligibility signal: it marks exactly the cases where the internal reader and the output disagree, which is where self-teaching has something to teach. For Thesis 4, it is a small working instance of a *frontier detector*: a signal, computed from the system's own internals, that it has moved onto ground its training did not cover — the cue an open-ended learner needs to route a task to acquisition and fresh evaluation rather than to confident answering.

8.7 Finding 6: the program already runs the Continuum loop by hand

Magic's research method is itself the admission principle in human-supervised form. Every idea is recorded with an explicit "kills it" condition and the smallest experiment that could run on one machine; every experiment leaves a result note, including failures; and two different AI models act as proposer and adversary, with several claims withdrawn or reduced under written cross-examination (Eidara Continuum, 2026e). The record shows the gate working: experiment 07's near-perfect lookup was downgraded when a control showed that most renamings turned unseen pairs into seen ones; experiment 06's 86% was downgraded after it proved tuned on a single seed; and the capability claim behind experiment 11's pointer-plus-calculator (100% in all conditions) was withdrawn when a plain transformer with the same supervision and longer training matched it, leaving a narrower efficiency claim. Proposal, adversarial testing, and refusal to commit an unearned claim are exactly the stages of Figure 1. The Continuum's research programme is to move those stages, one at a time, from the researchers into the system.

Scope of the internal evidence. These are results on a finite binding task with a model of about 72,000 parameters. Checks for this edition recomputed every experiment-13 summary from archived JSON, ran a finite-difference gradient spot check (largest sampled discrepancy 8.58×10^{-11}), traced pseudo-target construction and label access in the source, and evaluated the five checkpoints on a freshly seeded sample. Baseline training and the full adaptation run were not re-run for this edition; the internal review reports fresh reruns of selected experiments, which remains in-house verification, not external replication. The archived curves reuse evaluation samples and therefore do not satisfy the fresh-admission protocol of Theorem 3. Evaluation samples and the A/B directions share checkpoints, so neither questions nor directions are independent training replications, and equal optimizer steps in the ordinary-only control do not fully match teacher-forward computation or loss weighting. Magic does not demonstrate a ternary interpreter, dependency-aware memory, procedure discovery, or improvement of ψ. The archived notebooks for experiments 15–17 contain designs and predictions, not completed outcome tables; results the internal review reports for them, and its novelty-alarm figures, are labelled as reported rather than archived. The review itself is model-assisted in-house verification, and some of its headline figures (for example "53% to 99%") describe selected runs; this paper uses the five-seed archived means wherever they exist.

8.8 What the evidence supports, mapped to the theses

Thesis	Magic's contribution	Still required
V: learning from one's own evidence	Measured: reader beats output; self-teaching matches true labels and transfers	Open-world evidence; non-copying answers; standard architectures with a residual gap
III: governance makes gains stick	Measured: an 8–17-point transient valley before transfer motivates isolation; no gate was run	A running gate on fresh tasks, compared with ungated updating
II: improving the improver	Process only: the propose–attack–kill research loop runs with humans and two AI models in it	An admitted change to ψ that raises $M(\psi)$ on withheld families
I and IV: rate and frontier	Reported: an internal novelty alarm separates never-seen from trained cases (92–96% vs. 0%)	Acquisition-rate curves across expanding task families; the alarm used to route tasks to acquisition

9 Position in the 2025–2026 Landscape

The ingredients of this architecture each have predecessors, and the paper claims none of them individually. DreamCoder grows reusable symbolic abstractions with a learned search guide (Ellis et al., 2021); Gödel machines formalize proof-justified self-rewrites (Schmidhuber, 2003); learned optimizers train the update rule itself (Andrychowicz et al., 2016). STOP improves a model-calling program while leaving the model fixed (Zelikman et al., 2024), and the Darwin Gödel Machine evolves an archive of empirically evaluated self-modifying agents (Zhang et al., 2025). AlphaEvolve pairs language models with automated evaluators in an evolutionary loop and has improved infrastructure used in training the models that drive it (Novikov et al., 2025). SEAL trains a model to write its own fine-tuning data and update directives, rewarded by the updated model's downstream performance (Zweiger et al., 2025). AIDE[2] reports seven successive accepted improvements to a research agent's

own code in an autonomous eight-day run, with gains that generalize to four held-out benchmarks (Srikanth et al., 2026) — the closest current precedent for measured improvement of an acquisition procedure, at the level of a research harness rather than base-model weights. In the other direction, shadow evaluations in which the original authors grade agents' attempts at unpublished research questions find strong engineering but little progress on the questions themselves (Kirgis et al., 2026): bounded engineering gains and difficulty with open-ended research can both be true. On the learning-component side, Magic's mechanism sits beside work on context-versus-weight associations (Bietti et al., 2023; Olsson et al., 2022), pointer networks (Vinyals et al., 2015), context-aware decoding (Shi et al., 2023), test-time training (Sun et al., 2020), and context distillation (Snell et al., 2022).

Against that landscape, the Continuum's distinctive bet is a particular *combination*:

1. **A protected contract** that the self-improving system cannot rewrite, so every level of the system — including ψ — can change while the definition of improvement cannot.
2. **Anchored retention**, so a long chain of acceptable steps cannot drift silently away from the original self (Corollary 4b).
3. **Persistent, dependency-aware memory and lineage**, so the system's history of changes is itself auditable, replayable, and revocable.
4. **An evidence-reading teacher** that lets the system learn from material it encounters without external labels, with a stated bound on inherited error (Theorem 5).
5. **An explicit frontier requirement** (Thesis 4): the architecture treats the governed growth of its evaluated task frontier as a first-class obligation rather than assuming recursion can continue on a fixed benchmark — while forbidding the learner from redefining success.

Whether this combination outperforms simpler alternatives is an empirical question; Section 10 specifies how to answer it.

10 The Test Program

10.1 Architecture-level experiments (prospective)

Experiment	What the system must do	Result that refutes the target
A. Memory and correction	Retain scoped facts across sessions; invalidate dependent conclusions after a change; survive save/reload	Reset-memory or full-transcript controls match it at equal storage and retrieval budget
B. Procedure acquisition	Infer executable finite-domain transforms; reuse them in withheld compositions and longer inputs	Gains vanish on new compositions, or a fixed library matches them at equal total cost
C. Admission and retention	Run repeated candidate cycles including harmful and delayed-benefit candidates; check anchors	No reduction in damaging commits, or evaluation overhead erases the efficiency gain
D. Improving the improver	Allow bounded revisions to search, ranking, or target construction; compare old and new ψ from matched resets on new families	No held-out meta-utility gain once data, model calls, and compute are matched
E. Codon representation	Run identical semantics with ternary and binary/integer encodings	No reliable advantage attributable to the ternary alphabet

A concrete minimal suite uses three independently generated families: *records* (devices with versioned settings, where a new observation must invalidate only dependent predictions); *procedures* (finite-sequence transforms such as modular arithmetic, reversal, filtering, and composition, with selected compositions and lengths withheld); and *planning* (small transition systems in which one local rule changes). The evaluator computes answers from a separate implementation; acquisition, development, admission, and final-test data use separate seeds. Ablations remove one ingredient at a time — reset memory, transcript memory without dependencies, fixed library, acquisition without reuse, automatic acceptance, syntax-only checking, fixed ψ, alternative encodings — and a simple checkpoint-and-rollback store tests whether the full continuity machinery earns its cost.

10.2 Magic-to-architecture experiments (prospective)

Hypothesis	**Test**	**Refuting result**
H1: self-teaching adds value on a standard model	Normalized four-head model, random balanced pair splits, first-question primary score, at least 20 fresh seeds; target $\geq$2-point gain over matched ordinary adaptation	Upper confidence limit below 2 points, or retention fails
H2: improvement transfers	A pair group and task variants kept out of adaptation and selection; same target	Gains confined to exposed pairs
H3: admission improves repeated cycles	20 streams of 100 candidate cycles: ungated, syntax-only, and independent admission; count anchor losses over 1 point	No reduction in such losses at matched budget
H4: the improver improves	Restricted change to ψ; matched resets; fresh task families; fixed resources	No positive held-out meta-utility gain once compute and information are matched

10.3 The decisive improving-the-improver experiment

Freeze an initial ψ_0, a starting library and learner, and a development set of task families. Let the system propose a bounded revision ψ_1 to search ranking, replay choice, or target construction, selecting it with development and admission data only. Compare ψ_0 and ψ_1 from matched resets on previously withheld families at the same per-run budget, with the same base models, tools, observations, and starting capabilities; then charge the entire proposal-and-selection process in the total-cost comparison of Definition 1. The primary target is a positive paired held-out meta-utility with anchored retention; a distinct efficiency target requires a lower fully accounted cost. Failed proposals and unfavourable families stay in the report, and if only per-run quality improves, that narrower result is what is claimed. Repeating the procedure with new, independent final families — and seeing the gains persist and broaden — is what would turn one improvement of the improver into evidence of recursive self-improvement.

Stress tests run alongside: corrupt evidence, remove answer-bearing context, vary teacher coverage, and introduce contradictory or stale records; compare frozen and online teachers, true targets, normal-output self-training, copying without training, and compute-matched rehearsal. Retention checks include cases where the correct action is to revise a past belief or defer, not merely repeat an old answer.

10.4 Decision rules

For a first confirmatory study, the architecture-level target is at least a five-point paired gain in withheld-composition accuracy over the strongest matched fixed-library baseline, with at most a one-point loss on the initial retention suite; a separate efficiency target is at least a 20% reduction in total resources to reach a predeclared accuracy. These are prospective research targets, not consequences of the theory. A target is rejected if the upper confidence bound falls below it and is inconclusive if the interval spans it. The replication unit is the independently initialized learner or stream, with at least twenty streams as an initial plan and a power analysis on separate pilot streams before the final sample is frozen. For Hypothesis H4 and Experiment D — the Level-4 claim at the heart of this paper — a positive result must survive matched resets, repeated seeds, equivalent prior libraries, and a budget that includes the cost of generating and selecting the new ψ.

11 What Success Would Mean

The Continuum's founding intuition was that a mind is not a snapshot. This paper turns that intuition into a theory with parts that can be built and checked: a protected definition of improvement; a mutable self that includes its own learning procedure; persistent, revocable memory; an admission gate that lets gains accumulate and stops errors from doing the same; an explicit requirement that the frontier of evaluated competence keep expanding; and a way for the system to teach itself from the evidence it reads. The theorems say what this recursion can guarantee and where its guarantees stop: bounded execution is not intelligence, an admitted gain is not universal progress, a fixed-margin ceiling is not a proof that new families are necessary, and a geometric series under an assumed premise is not a forecast. The Magic program shows the self-teaching mechanism working on a real, if small, system — reading better than it speaks, learning from its own reading, transferring to cases it never saw, and passing through a valley that isolated exploration is designed to make safe.

If the program in Section 10 succeeds — if admitted revisions to the acquisition procedure keep raising the retained acquisition rate, after retention and full cost accounting, on task families that played no part in choosing them — the Continuum will have demonstrated a bounded, governed form of recursive self-improvement: not an intelligence explosion, but a measurable, auditable climb in the rate at which a system gets better at getting better. That is what Thesis 1 calls a path toward generality. If it fails, the same program will identify which commitment did not earn its cost. Either outcome advances the question the Continuum was built to ask.

A One Complete Cycle

1. Load a versioned stable state and protected contract K; verify artifact identity and dependencies; initialize the shared budget, including timeouts for external primitives.
2. Ingest the next input as attributed evidence; retrieve only records and procedures whose scope and dependency versions apply; keep disputed claims apart from trusted assertions.
3. Execute within fuel, stack, storage, and external-call limits; record outputs, failures, and proposed changes. Appending an observation is not itself a claim of improvement.
4. Use ψ to propose a bounded change to memory rules, P, θ, or ψ; create a separate branch; declare changed fields and effects; record all data and compute.

5. Train or search within the exploration budget; select a checkpoint by a predeclared rule; freeze the candidate before admission tasks are drawn.
6. Evaluate candidate, stable state, and initial anchors on fresh evaluator-held tasks, including invalid-input behaviour and total resources; apply the admission rule of Theorem 3 or another valid rule.
7. Commit an accepted change atomically with its tests and ancestry; on rejection keep stable operation and a marked exploratory record; revalidate dependants of any changed dependency.
8. Check save/reload and replay at planned intervals; continue only within candidate and evaluation budgets. Treat any change to K as a new contract.

B Verification Ledger

Item	Check	Status
Theorem 3 radius	$r = \sqrt{2\ln(300/0.05)/200{,}000} \approx 0.009327$	Recomputed
Corollary 4	Summation, anchor, and acceptance-count bounds	Proof checked
Proposition 8	Two tapes enumerated; equal histograms, $H_3 = 1$, outputs 1 and 0	Enumerated
Theorem 9 example	$f(x) = 2(x+1) \bmod 8$ on all eight inputs; costs $2r$ vs. $r+3$	Exhaustive
Theorem 5	Indicator inequality on all binary triples	Enumerated
Proposition 2	Finite geometric sum under the stated premise	Algebra checked
Magic, experiment 13	Means, endpoint gains ($80.08 - 56.06$, $82.98 - 62.65$), teacher counts 9,988/10,000; all five curve readings	Recomputed from archived JSON
Magic, gradient	Largest sampled finite-difference discrepancy 8.58×10^{-11}	In-house check
Magic, probe	256 fresh episodes per checkpoint, seed 20261118; causal-mask equivalence passes	In-house check

The inspected Magic archive is identified by SHA-256
`82fd36e5daeb0004b80765cfe06e41491b502cd9068da465c88d23cf25fc38e7`.
Shared episode samples, reused checkpoints, and within-episode questions are not counted as independent training replications.

References

Andrychowicz, M., Denil, M., Gomez, S., Hoffman, M. W., Pfau, D., Schaul, T., Shillingford, B., and de Freitas, N. (2016). Learning to learn by gradient descent by gradient descent. *NeurIPS 29*. arXiv:1606.04474.

Bietti, A., Cabannes, V., Bouchacourt, D., Jégou, H., and Bottou, L. (2023). Birth of a transformer: A memory viewpoint. *NeurIPS 36*. arXiv:2306.00802.

Chen, M., Wang, L., and Qu, B. (2026). Recursive self-improvement in AI: From bounded self-refinement to autonomous research loops. arXiv:2607.07663.

Dwork, C., Feldman, V., Hardt, M., Pitassi, T., Reingold, O., and Roth, A. (2015). Generalization in adaptive data analysis and holdout reuse. arXiv:1506.02629.

The Eidara Continuum (2026a). *The Eidara Continuum Bible*. Internal reference edition, §§3.5, 3.7, 3.8, 4.6, 4.7. Architectural provenance.

The Eidara Continuum (2026b). *The Continuum Doctrine*. Internal canonical work. Philosophical provenance.

The Eidara Continuum (2026c). Internal historical monograph of the Continuum. Historical provenance; restricted title omitted.

The Eidara Continuum (2025–2026). Internal kernel specifications, architectural blueprints, and runtime notes, including the September 2025 minimal kernel and October 2025 blueprint.

The Eidara Continuum (2026d). *Magic: Experimental source, checkpoints, and research notebooks*. In-house archive, 17 November 2026; experiments 01–17.

The Eidara Continuum (2026e). *The "Magic" program — comprehensive technical review*. Internal review and testing summary, 18 November 2026. In-house verification; not external replication.

Ellis, K., Wong, C., Nye, M., Sablé-Meyer, M., Cary, L., Morales, L., Hewitt, L., Solar-Lezama, A., and Tenenbaum, J. B. (2021). DreamCoder: Bootstrapping inductive program synthesis with wake-sleep library learning. *PLDI 2021*. arXiv:2006.08381.

Everitt, T., Hutter, M., Kumar, R., and Krakovna, V. (2021). Reward tampering problems and solutions in reinforcement learning: A causal influence diagram perspective. *Synthese*. arXiv:1908.04734.

Kirgis, P., Kapoor, S., Schwartz, A., Rabanser, S., et al. (2026). Can AI agents conduct open-ended AI research? Early evidence from two case studies. arXiv:2607.27191.

Hoeffding, W. (1963). Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30.

Novikov, A., et al. (2025). AlphaEvolve: A coding agent for scientific and algorithmic discovery. Google DeepMind. arXiv:2506.13131.

Olsson, C., et al. (2022). In-context learning and induction heads. *Transformer Circuits Thread*.

Schmidhuber, J. (2003). Gödel machines: Self-referential universal problem solvers making provably optimal self-improvements. arXiv:cs/0309048.

Shi, W., Han, X., Lewis, M., Tsvetkov, Y., Zettlemoyer, L., and Yih, W. (2023). Trusting your evidence: Hallucinate less with context-aware decoding. arXiv:2305.14739.

Shumailov, I., Shumaylov, Z., Zhao, Y., Papernot, N., Anderson, R., and Gal, Y. (2024). AI models collapse when trained on recursively generated data. *Nature*, 631:755–759.

Snell, C., Klein, D., and Zhong, R. (2022). Learning by distilling context. arXiv:2209.15189.

Srikanth, D., Zhao, B., Xu, D., Wu, Y., and Jiang, Z. (2026). Recursive self-improvement of AI research agents. Weco AI. arXiv:2609.26457.

Sun, Y., Wang, X., Liu, Z., Miller, J., Efros, A. A., and Hardt, M. (2020). Test-time training with self-supervision for generalization under distribution shifts. *ICML*, PMLR 119:9229–9248.

Vinyals, O., Fortunato, M., and Jaitly, N. (2015). Pointer networks. *NeurIPS 28*. arXiv:1506.03134.

Zelikman, E., Lorch, E., Mackey, L., and Kalai, A. T. (2024). Self-Taught Optimizer (STOP): Recursively self-improving code generation. *COLM 2024*. arXiv:2310.02304.

Zhang, J., Hu, S., Lu, C., Lange, R., and Clune, J. (2025). Darwin Gödel Machine: Open-ended evolution of self-improving agents. arXiv:2505.22954.

Zweiger, A., Pari, J., Guo, H., Akyürek, E., Kim, Y., and Agrawal, P. (2025). Self-adapting language models. *NeurIPS 38*. arXiv:2506.10943.