

# Hindsight Hierarchies for Reasoning Models: A Quantitative Yield and Trajectory Analysis of the Reverse-Engineering Self-Improvement Loop

## Hindsight Hierarchies for Reasoning Models: A Quantitative Yield and Trajectory Analysis of the Reverse-Engineering Self-Improvement Loop

### Abstract

---

Reasoning models for formal mathematics hit a data wall: training corpora consist of problems paired with machine-checked solutions, so problems beyond a model's current ability yield no solved examples, and the corpus stops growing exactly where improvement is needed. The hindsight-hierarchies proposal of arXiv:2610.12168 addresses this by jointly training one model to predict solution ideas from problems alone, to reverse-engineer ideas from problems with supplied solutions, and to solve problems using provided ideas; however, that work supplies a formal specification and a Lean instantiation while leaving empirical evaluation to future work, so the loop's data economics are unquantified. This paper provides the missing quantitative layer. We formalize the loop as a discrete dynamical system over a solved-problem corpus and an idea-supervision corpus, parameterized by ideas per solution  $m$ , extraction recall  $r$ , extraction precision  $p$ , and solve rate  $\sigma$ . Under stated assumptions ( $m = 5$ ,  $r = 0.6$ ,  $p = 0.8$ ,  $\sigma = 0.1$ , pool  $P = 10^4$ , seed corpus  $N_0 = 10^3$ ), we derive a per-round clean idea amplification  $A = mrp = 2.4$ , a trajectory reaching  $N_5 = 4685.59$  solved problems after five rounds, a cumulative 42791.26 clean idea pairs ( $42.79 \times$  the seed corpus), and a model-generated corpus fraction of 78.66% by round five, making extraction precision, not solve rate, the binding quantity; the break-even precision against direct fine-tuning on solutions is  $p > 1/3$ . The analysis converts an untested proposal into a falsifiable projection with explicit failure conditions.

### 1. Introduction

---

Teams building reasoning models for formal mathematics face a concrete bottleneck that grows worse as the models improve. Training data for such systems consists of problems paired with machine-checked solutions; a problem the current model cannot solve contributes no solution, hence no new supervision. The corpus therefore expands fastest where the model is already competent and stalls precisely at the difficulty frontier where capability gains are sought. Today this blocks the practical path of self-improvement for interactive theorem proving: a lab with a seed corpus of verified proofs and a large pool of unproved statements has no principled procedure for converting compute into new verified training examples at the frontier, and must instead rely on scarce human expert proofs, whose cost limits corpus growth. Every training iteration on self-generated data is capped by the fraction of the corpus the model can close unaided, and scaling compute does not manufacture solutions to problems beyond the current frontier. A procedure that reliably converts supplied solutions at any difficulty into reusable supervision would change a specific design decision: instead of paying experts to prove frontier problems, one pays them to solve a smaller sample and lets the training loop propagate the solution ideas, so that the frontier of automatically provable statements advances at a rate set by the supply of human-written proofs rather than by the model's current solve rate.

The hindsight-hierarchies proposal [1], [2] is a candidate for exactly this role. Its central observation is that even when a problem exceeds the model's solving ability, a supplied solution enables the model to extract useful solution ideas in hindsight. It operationalizes this by jointly training the same model on three capabilities: predicting solution ideas from problems alone, reverse-engineering ideas from problems with known solutions, and solving problems

using provided ideas. The loop alternates between reverse-engineering ideas from problems with supplied solutions and using those ideas as additional supervision for joint training of all three capabilities. The authors give a formal specification and a concrete instantiation for interactive theorem proving in the Lean theorem prover, and state explicitly that empirical evaluation remains future work.

What is missing, and what blocks any decision to invest in building the loop, is a quantitative account of its data economics. Without one, a practitioner cannot answer the questions that determine whether the loop is worth building: how much new supervision does one round yield per unit of supplied solution, at what extraction quality does the loop help rather than poison its own training signal, and how fast does the solved-problem corpus grow as a function of the solve rate on previously unsolved problems? This paper supplies that account analytically. We model the loop as a discrete dynamical system, derive its growth law in closed form, compute a worked trajectory under explicitly stated parameter assumptions, identify which parameter the result is most sensitive to, and state the measurement that would confirm or refute the model. All numerical results are projections under stated assumptions, not measurements; their role is to make the proposal falsifiable before any training run is funded.

Contributions:

- A dynamical-systems formalization of the hindsight-hierarchy loop over two coupled corpora (solved problems and idea-supervision pairs), with the growth law  $N_{t+1} = N_t + \sigma(P - N_t)$  and its closed-form solution  $N_t = P - (P - N_0)(1 - \sigma)^t$ , derived in Section 3 and Section 4.
- A derivation of the clean idea-amplification factor  $A = mrp$  and the cumulative amplification over five rounds, computed as  $A = 2.4$  per round and  $42.79\times$  cumulative under stated assumptions, in Section 4.
- A break-even condition against direct fine-tuning on solutions,  $mrp > 1$ , giving a break-even precision of  $p > 1/3$  at the stated  $m$  and  $r$ , derived in Section 4.
- Identification of the noise-accumulation boundary: the model-generated corpus fraction reaches 78.66% by round five under the same assumptions, implying extraction precision  $p$  is the binding parameter, derived in Section 4 and analyzed in Section 6.
- A translation of these results into engineering guidance: which parameter to measure first, what the smallest de-risking experiment is, and which constraints bind first, in Section 7.

## 2. Background and Related Work

---

The direct object of this analysis is the hindsight-hierarchies paper itself, which appears in the bibliography as two entries with identical content. Entry [1] supplies the full abstract: the self-improvement loop rests on the observation that a supplied solution can enable extraction of useful solution ideas even when the problem exceeds the model's current solving ability; the method jointly trains one model on three capabilities (predicting ideas from problems alone, reverse-engineering ideas from problems and known solutions, and solving problems using provided ideas); the loop alternates reverse-engineering with joint retraining; and the paper gives a formal specification plus a Lean instantiation while stating that empirical evaluation remains future work. Entry [2] is the same work under its arXiv identifier 2610.12168v1, with the abstract truncated at the same point; we cite [2] when referring to the work by identifier and [1] when quoting its stated scope. Because the supplied abstracts state no empirical numbers, every quantitative claim about the loop in this paper is our own derivation, not a report of a result from [1] or [2].

The closest methodological ancestor in the supplied bibliography is hindsight reinforcement learning. Entry [3], "Hierarchical Reinforcement Learning with Hindsight" (arXiv:1805.08180v2), states that reinforcement learning algorithms can suffer from poor sample efficiency when rewards are delayed and sparse, and introduces a solution enabling agents to learn temporally extended actions at multiple levels of abstraction in a sample-efficient and automated fashion, combining universal value functions with hindsight learning so that agents learn policies

belonging to different levels. The shared idea with [1] is the relabeling of past experience as supervision: what was an unsuccessful attempt at one goal becomes a successful demonstration of a hindsight goal. Our analysis imports this framing but differs in the object relabeled: [3] relabels achieved states as goals, while [1] relabels supplied solutions as sources of transferable ideas, which introduces the extraction-precision parameter  $p$  that has no analogue in the hindsight-goal setting and that our sensitivity analysis in Section 4 identifies as binding. The supplied summary of [3] reports no quantitative results, so we draw only the structural correspondence.

A second control-theoretic antecedent is entry [7], "Learning from the Hindsight Plan -- Episodic MPC Improvement" (arXiv:1609.09001v2). Its abstract states that model predictive control performs re-planning at every time step with a limited horizon due to computational and real-time constraints and often for robustness to potential model errors, that the limited horizon leads to suboptimal performance, and that the work considers learning from the hindsight plan; the abstract truncates before specifying the mechanism. The connection to our argument is structural: both [7] and the loop of [1] treat the output of a planning or solving episode as an input to improving the next episode, a self-improvement loop whose convergence properties depend on the quality of the reused artifact. The supplied summary of [7] gives no further detail, so we use it only to establish that hindsight-reuse loops have a precedent in control, not to import any specific result.

On the data-supply side, the loop competes with and complements procedural generation. Entry [9], "Reasoning Core: A Scalable Procedural Data Generation Suite for Symbolic Pre-training and Post-Training" (arXiv:2603.02208v1), states that training on verifiable symbolic data is a promising way to expand the reasoning frontier of language models beyond standard pre-training corpora, that existing procedural generators often rely on fixed puzzles or templates and do not deliver the distributional breadth needed at scale, and that Reasoning Core procedurally generates verifiable symbolic reasoning data. This matters because procedural generation and hindsight extraction are complementary supply channels for verifiable symbolic supervision: [9] generates problems from templates, while the loop of [1] converts solved problems into idea supervision. Our corpus-growth model in Section 3 treats the problem pool  $P$  as exogenous; a generator of the kind [9] describes is one mechanism that could enlarge  $P$ , and we flag the coupling as an open question rather than modeling it.

Entry [8], "Transfer Learning with Pre-trained Conditional Generative Models" (arXiv:2204.12833v3), states that transfer learning is crucial for training deep neural networks on new target tasks, that current methods always assume at least one of overlapping source and target label spaces, available source datasets, or consistent target and source architectures, and that holding these assumptions is difficult in practical settings because the target task raises difficulties the abstract truncates before specifying. Its relevance is that the three jointly trained capabilities in [1] share one model across tasks whose label spaces differ (problems, problem-solution pairs, problem-idea pairs), so the transfer-learning framing of [8] raises the question of whether joint training transfers positively across the three capabilities; the supplied summary states no finding on this, and we treat positive transfer as an assumption of our model, listed in Section 6.

Entry [6], "Why & When Deep Learning Works: Looking Inside Deep Learnings" (arXiv:1705.03921v1), states that the Intel Collaborative Research Institute for Computational Intelligence asked six leading deep-learning researchers to address the challenge of why and when deep learning works, with the goal of looking inside deep learning and providing insights on how deep networks function; the abstract truncates there. We cite it as evidence that interpretive questions about why a training procedure succeeds remain open at the field level, which supports our decision to give the loop of [1] a falsifiable analytical model rather than a mechanistic claim about why joint training of the three capabilities would work.

Two entries concern reasoning formalisms and human factors adjacent to, but distinct from, the loop. Entry [5], "Reasoning with Justifiable Exceptions in Contextual Hierarchies (Appendix)" (arXiv:1808.01874v1), states that it is an appendix to the paper by Bozzato, Serafini and Eiter (2018) providing further details on the language, the

complexity results, and the Datalog translation introduced in the main paper; the supplied summary gives no further detail on those results. We note the work because its subject, hierarchical reasoning with exceptions, is terminologically adjacent to "hindsight hierarchies" while being a symbolic-representation line of work rather than a training method, and we draw no technical content from it. Entry [4], "Navigating the State of Cognitive Flow: Context-Aware AI Interventions for Effective Reasoning Support" (arXiv:2504.16021v1), states that flow theory describes an optimal cognitive state of deep focus and intrinsic motivation when task difficulty aligns with skill level, that interventions disrupting cognitive flow can hinder rather than enhance decision-making in AI-augmented reasoning, and that the paper proposes a context-aware cognitive augmentation framework adapting interventions based on context. Its relevance is a design caution for human-facing deployment: if the loop is used to select which problems and hints to show a human mathematician, supplying ideas for problems far above the user's skill level is precisely the difficulty-skill mismatch the flow framing warns against. The supplied summary reports no empirical measurements, so this remains a caution, not a quantified constraint.

Finally, four QNFO corpus entries appear in the bibliography. Entry [13], "Structured Cross-Domain Translation as a Testable Consilience Mechanism: Protocol Design, Truth-Preservation Conditions, and a Computational Evaluation Blueprint," states that interdisciplinary analogy-making is usually treated as an ad hoc rhetorical act rather than a reproducible research procedure, and that the paper formalizes a conjecture that systematically translating a single formal object into multiple domain lexicons is such a procedure, with protocol design, truth-preservation conditions, and a computational evaluation blueprint. This is methodologically parallel to the loop of [1] at the level of representation transport: both treat the translation of a solution artifact into a reusable intermediate representation as the unit of supervision, and [13]'s truth-preservation framing supports treating idea extraction as a checkable operation rather than a free-text summary. Entries [10] ("QNFO: Syntactic Generation"), [11] ("QNFO: Impact of Cognitive Linearity on Epistemic Modeling"), and [12] ("QNFO: Projective Geometric Frameworks for Semantic Structures") are supplied with no summary text; we record that the supplied entries give no further detail, and relate them to our argument only through their titles, as candidate representation frameworks within which "solution ideas" might be formalized.

In summary, the literature supplied establishes the hindsight-reuse precedent in reinforcement learning and control [3], [7], the verifiable-symbolic-data substrate [9], open questions about transfer and mechanism [8], [6], and adjacent formal, cognitive, and translation framings [4], [5], [13]. None of the supplied entries quantifies the data economics of the loop of [1]; that is the gap this paper fills.

### 3. Methods

---

#### 3.1 Objects and notation

We model the loop of [1] as acting on two coupled corpora. Let:

- $P \in \mathbb{N}$ : the total pool of problems available to the system, fixed within the analysis horizon.
- $N_t \in \mathbb{N}$ : the number of problems with verified solutions at the start of round  $t$ , with  $N_0$  the seed corpus.
- $U_t = P - N_t$ : the number of unsolved problems at round  $t$ .
- $m \in \mathbb{N}$ : the mean number of extractable solution ideas per solution.
- $r \in [0, 1]$ : extraction recall, the fraction of the  $m$  ideas in a supplied solution that the reverse-engineering capability recovers.
- $p \in [0, 1]$ : extraction precision, the fraction of recovered ideas that are genuinely useful (correct and relevant) rather than spurious.

- $\sigma \in [0, 1]$ : the per-round solve rate, the probability that the model, equipped with ideas predicted by the idea-prediction capability, solves a previously unsolved problem and has the solution machine-verified.

"Solution idea" is the term of art of [1]; we operationalize it as any reusable intermediate representation extracted from a solution that can be supplied as conditioning to the solving capability. In the Lean instantiation of [1], a solution is a machine-checked proof, so a newly produced solution enters the corpus only if the proof checker accepts it; this verification gate is what keeps the solved-problem corpus free of unsound examples and is central to the noise analysis of Section 4.4.

### 3.2 The loop as a discrete dynamical system

One round of the loop, following the alternation stated in [1] (reverse-engineer ideas from problems with supplied solutions, then use these ideas as additional supervision for joint training of all three capabilities), induces the following update:

1. Idea extraction: from the  $N_t$  solved problems, extract idea-supervision pairs at rate  $mr$  per solution, with precision  $p$ .
2. Joint retraining: retrain the single model on the three capabilities using the accumulated idea pairs and solved problems.
3. Solving: deploy the retrained model on the  $U_t$  unsolved problems with predicted ideas as conditioning; a fraction  $\sigma$  of attempts succeeds and passes machine verification, adding  $\sigma U_t$  new solved problems.

The solved-problem corpus therefore obeys:

$$N_{t+1} = N_t + \sigma(P - N_t).$$

The idea-supervision corpus accumulates across rounds. Let  $M_t$  denote the cumulative number of extracted idea pairs after round  $t$ :

$$M_t = \sum_{\tau=0}^t N_{\tau} mr,$$

of which the clean fraction is  $p$ , so the cumulative clean idea pairs are:

$$C_t = p M_t = \sum_{\tau=0}^t N_{\tau} mrp.$$

### 3.3 Closed-form growth law

The update equation is affine in  $N_t$ . Define the deficit  $D_t = P - N_t$ . Then:

$$D_{t+1} = P - N_{t+1} = P - N_t - \sigma(P - N_t) = (1 - \sigma)D_t,$$

so  $D_t = D_0(1 - \sigma)^t$  with  $D_0 = P - N_0$ , giving:

$$N_t = P - (P - N_0)(1 - \sigma)^t. \tag{1}$$

This is the discrete logistic approach to saturation: growth is fastest at  $t = 0$  when the unsolved pool is largest, and the marginal yield per round decays geometrically at rate  $(1 - \sigma)$ .

### 3.4 Amplification factors and break-even

We define two amplification quantities. The per-round clean idea amplification per solved problem is:

$$A = mrp. \quad (2)$$

The cumulative clean idea amplification after  $T$  rounds relative to the seed corpus size  $N_0$  is:

$$A_T = \frac{C_T}{N_0} = \frac{mrp}{N_0} \sum_{\tau=0}^T N_\tau. \quad (3)$$

Substituting (1) and using the geometric sum  $\sum_{\tau=0}^T (1 - \sigma)^\tau = \frac{1 - (1 - \sigma)^{T+1}}{\sigma}$ :

$$\sum_{\tau=0}^T N_\tau = (T + 1)P - (P - N_0) \cdot \frac{1 - (1 - \sigma)^{T+1}}{\sigma}. \quad (4)$$

The break-even condition against direct fine-tuning on solutions follows from comparing amplification factors. Direct fine-tuning produces exactly 1 usable supervision example per machine-checked solution. The loop is worth its additional reverse-engineering cost only if its clean yield per solved problem exceeds 1:

$$A = mrp > 1 \quad \Longleftrightarrow \quad p > \frac{1}{mr}. \quad (5)$$

Equation (5) is evaluated numerically in Section 4.

### 3.5 Model-generated fraction

The fraction of the corpus at round  $t$  that was produced by the model itself (rather than supplied in the seed) is:

$$f_t = 1 - \frac{N_0}{N_t}. \quad (6)$$

This quantity governs error-compounding risk: although each new solution is machine-verified for soundness (the Lean gate), the ideas extracted from model-generated solutions inherit the idiosyncrasies of model-generated proofs, and the idea labels produced by reverse-engineering are noisy at rate  $1 - p$  regardless of provenance.

### 3.6 Assumption register

All numerical results below are projections under the following assumptions, chosen as illustrative mid-range values because the supplied abstracts of [1], [2] state no empirical numbers:  $P = 10^4$  problems in the pool, a plausible size for a formalized statement collection in one domain;  $N_0 = 10^3$  solved problems in the seed corpus, one tenth of the pool so that the loop has room to grow;  $m = 5$  extractable ideas per solution, a mid-range count for reusable intermediate steps in a proof;  $r = 0.6$  extraction recall, meaning the reverse-engineering capability recovers three of five ideas;  $p = 0.8$  extraction precision, meaning four of five recovered ideas are judged correct and relevant; and  $\sigma = 0.1$  per-round solve rate on previously unsolved problems, a conservative starting competence. Each value is an assumption, not a measurement, and Section 6 lists the sensitivity of the results to each.

## 4. Analysis

### 4.1 The growth trajectory

With  $P = 10^4$ ,  $N_0 = 10^3$ , and  $\sigma = 0.1$ , the deficit is  $D_0 = P - N_0 = 9000$  and decays as  $D_t = 9000 \cdot 0.9^t$ . Applying Equation (1),  $N_t = 10^4 - 9000 \cdot 0.9^t$ :

- $N_0 = 10^4 - 9000 \cdot 1 = 1000$ .
- $N_1 = 10^4 - 9000 \cdot 0.9 = 10^4 - 8100 = 1900$ .
- $N_2 = 10^4 - 9000 \cdot 0.81 = 10^4 - 7290 = 2710$ .
- $N_3 = 10^4 - 9000 \cdot 0.729 = 10^4 - 6561 = 3439$ .
- $N_4 = 10^4 - 9000 \cdot 0.6561 = 10^4 - 5904.9 = 4095.1$ .
- $N_5 = 10^4 - 9000 \cdot 0.59049 = 10^4 - 5314.41 = 4685.59$ .

Each round adds  $\sigma U_t = 0.1(10^4 - N_t)$  new solved problems: 900, 810, 729, 656.1, 590.49 in rounds 1 through 5 respectively, a geometric decay at rate  $1 - \sigma = 0.9$ . The round-five corpus of  $N_5 = 4685.59$  solved problems is  $4.69\times$  the seed corpus, computed as  $4685.59/1000 = 4.68559$ .

The number of rounds to reach a target corpus fraction  $\phi$  of the pool follows from  $(1 - \sigma)^t = (P - \phi P)/(P - N_0)$ . For  $\phi = 0.9$ :  $0.9^t = 1000/9000 = 1/9$ , so  $t = \ln(1/9)/\ln(0.9) = (-2.19722)/(-0.10536) = 20.85$ , i.e. 21 rounds. At  $\sigma = 0.1$ , saturation is slow: half the pool satisfies  $0.9^t = (P - 0.5P)/(P - N_0) = 5000/9000 = 5/9$ , so  $t = \ln(5/9)/\ln(0.9) = (-0.587787)/(-0.105361) = 5.5788$ , i.e. round 6.

### 4.2 Cumulative amplification

The per-round clean idea amplification is, from Equation (2):

$$A = mrp = 5 \times 0.6 \times 0.8 = 2.4.$$

Each machine-checked solution thus yields on average 2.4 clean idea-supervision pairs per round of extraction. The cumulative sum of solved problems over rounds 0 through 5 is, from Equation (4):

$$\sum_{\tau=0}^5 N_\tau = 6 \times 10^4 - 9000 \times \frac{1 - 0.9^6}{0.1}.$$

Since  $0.9^6 = 0.531441$ , we get  $\frac{1-0.531441}{0.1} = 4.68559$ , so:

$$\sum_{\tau=0}^5 N_\tau = 60000 - 9000 \times 4.68559 = 60000 - 42170.31 = 17829.69.$$

Cross-check by direct addition:  $1000 + 1900 + 2710 + 3439 + 4095.1 + 4685.59 = 17829.69$ , matching. The cumulative clean idea pairs are:

$$C_5 = mrp \sum_{\tau=0}^5 N_\tau = 2.4 \times 17829.69 = 42791.256 \approx 42791.26.$$

The cumulative amplification relative to the seed corpus is, from Equation (3):

$$A_5 = \frac{C_5}{N_0} = \frac{42791.256}{1000} = 42.791256 \approx 42.79 \times .$$

Five rounds of the loop therefore manufacture the equivalent of roughly 42.79 seed corpora of clean idea supervision, under the stated assumptions.

#### 4.3 Break-even against direct fine-tuning

Direct fine-tuning on machine-checked solutions yields exactly 1 usable supervision example per solution. From Equation (5), the loop beats this when  $m r p > 1$ . At  $m = 5$  and  $r = 0.6$ :

$$p > \frac{1}{m r} = \frac{1}{5 \times 0.6} = \frac{1}{3} \approx 0.3333.$$

The stated  $p = 0.8$  exceeds this threshold by a factor of 2.4, since  $A = 2.4 > 1$ . The margin is the ratio  $A/1 = m r p$ , so the loop tolerates a precision as low as 0.3333 before its clean yield per solution falls below direct fine-tuning. Equivalently, at  $p = 0.8$  the loop tolerates recall as low as  $r > 1/(m p) = 1/4 = 0.25$ , computed as  $1/(5 \times 0.8)$ .

#### 4.4 Noise accumulation and the model-generated fraction

From Equation (6), the model-generated corpus fraction at round 5 is:

$$f_5 = 1 - \frac{N_0}{N_5} = 1 - \frac{1000}{4685.59} = 1 - 0.213418 = 0.786582 \approx 78.66\%.$$

The intermediate values are  $f_1 = 1 - 1000/1900 = 0.473684$ ,  $f_2 = 1 - 1000/2710 = 0.631000$ ,  $f_3 = 1 - 1000/3439 = 0.709217$ ,  $f_4 = 1 - 1000/4095.1 = 0.755802$ . The fraction crosses 50% already at round 1 under these assumptions, because the first round adds 900 model-verified problems against a seed of 1000.

The noisy-idea inflow per round is  $(1 - p) m r N_t = 0.2 \times 3 \times N_t = 0.6 N_t$  spurious pairs, against  $2.4 N_t$  clean pairs; the instantaneous clean-to-noise ratio is fixed at  $p/(1 - p) = 4$  regardless of round. The risk is therefore not a rising noise fraction within the idea stream but the sheer cumulative volume: by round 5 the corpus contains  $(1 - p) m r \sum_{\tau=0}^5 N_\tau = 0.6 \times 17829.69 = 10697.81$  spurious pairs alongside 42791.26 clean ones, and the training set is 78.66% model-generated, so any systematic idiosyncrasy of model-generated proofs is amplified, not diluted, as the loop runs.

#### 4.5 Sensitivity ranking

The amplification  $A = m r p$  is linear in each of  $m$ ,  $r$ ,  $p$ , so a relative change  $\Delta x/x$  in any one parameter changes  $A$  by the same relative amount. The binding-parameter question is therefore about which parameter is least controllable in practice, not about local sensitivity of  $A$ . Two considerations rank  $p$  first. First, the break-even threshold  $p > 1/3$  lies far below the assumed 0.8, but  $p$  is the only parameter that cannot be measured from the corpus alone:  $m$  and  $r$  are measurable by sampling supplied solutions and checking recovered ideas against them, while  $p$  requires judging relevance of extracted ideas, a noisier measurement. Second, the corpus-growth rate  $\sigma$  does not enter  $A$  at all; it enters only the trajectory timescale through  $(1 - \sigma)^t$ . A loop with  $\sigma = 0$  still produces  $A = 2.4$  clean pairs per solved problem per round from the seed corpus; a loop with  $p < 1/3$  produces a clean yield below direct fine-tuning regardless of how fast it solves new problems. Extraction precision, not solve rate, is the binding quantity.

## 5. Results

All numbers in this section are projections computed in Section 4 under the assumption register of Section 3.6 ( $P = 10^4$ ,  $N_0 = 10^3$ ,  $m = 5$ ,  $r = 0.6$ ,  $p = 0.8$ ,  $\sigma = 0.1$ ); none is an empirical measurement.

- **Growth law.** The solved-problem corpus follows  $N_t = P - (P - N_0)(1 - \sigma)^t$ , reaching  $N_5 = 4685.59$  solved problems after five rounds, i.e.  $4.69\times$  the seed corpus, with per-round additions decaying geometrically: 900, 810, 729, 656.1, 590.49.
- **Saturation timescale.** At  $\sigma = 0.1$ , the corpus reaches half the pool at round 6 ( $t = 5.58$ ) and 90% of the pool at round 21 ( $t = 20.85$ ).
- **Per-round amplification.** The clean idea amplification is  $A = mrp = 2.4$  clean idea pairs per solved problem per round.
- **Cumulative amplification.** Over five rounds, the loop yields  $C_5 = 42791.26$  cumulative clean idea pairs, an amplification of  $A_5 = 42.79\times$  the seed corpus size.
- **Break-even.** The loop beats direct fine-tuning on solutions when  $mrp > 1$ ; at  $m = 5$ ,  $r = 0.6$  this is  $p > 1/3 \approx 0.3333$ , and the assumed  $p = 0.8$  clears it by a factor of 2.4. Symmetrically, at  $p = 0.8$  the loop tolerates  $r > 0.25$ .
- **Noise exposure.** The model-generated corpus fraction reaches 78.66% by round five ( $f_1 = 47.37\%$ ,  $f_2 = 63.10\%$ ,  $f_3 = 70.92\%$ ,  $f_4 = 75.58\%$ ), and the cumulative spurious-idea volume is 10697.81 pairs against 42791.26 clean pairs, a fixed clean-to-noise ratio of 4.
- **Binding parameter.** Extraction precision  $p$  is the binding quantity: it alone determines whether the loop's clean yield exceeds direct fine-tuning, it is the hardest parameter to measure from the corpus alone, and the solve rate  $\sigma$  affects only the timescale, not the yield per solution.

## 6. Discussion

The central result is that the loop of [1] is economically viable under a wide band of extraction quality but is governed by a single quantity,  $A = mrp$ , that a practitioner can measure cheaply before committing to the full loop. The strongest reading of the projection is the break-even threshold: because  $p > 1/3$  is a low bar, the plausible failure mode of the loop is not marginal economics but a qualitatively different one, in which extracted "ideas" are so generic that supplying them does not condition the solver at all, in which case the operationalization of "idea" rather than the number  $p$  is wrong. That failure would not show up as  $p$  slightly below threshold; it would show up as the idea-conditioned solve rate  $\sigma$  being indistinguishable from the unconditioned solve rate, which is a measurement the smallest experiment of Section 7 is designed to make.

**Falsification.** The main result, that the loop amplifies supervision by  $A = mrp > 1$  per solved problem, is falsified by a single measurement: sample  $n$  supplied solutions, run the reverse-engineering capability, have experts judge each recovered idea for correctness and relevance, and compute the empirical  $\hat{A} = m\hat{r}\hat{p}$ . If  $\hat{A} \leq 1$  at plausible parameter values (for example,  $\hat{m} = 2$ ,  $\hat{r} = 0.4$ ,  $\hat{p} = 0.7$  gives  $\hat{A} = 0.56$ ), the loop's clean yield per solution is below direct fine-tuning and the amplification claim fails. A second falsifier is dynamic: if the idea-conditioned solve rate on held-out unsolved problems equals the unconditioned rate, the loop's third capability is inert and the growth law's  $\sigma$  term collapses to the unconditioned rate, removing the loop's advantage over simply collecting solutions.

**Limitations.** The projection rests on the following assumptions, each with its sensitivity:

- **Constant parameters.**  $m$ ,  $r$ ,  $p$ ,  $\sigma$  are held fixed across rounds. If  $p$  degrades as the corpus becomes model-dominated ( $f_5 = 78.66\%$ ), the clean yield decays round over round and the cumulative figure  $42.79\times$  is an

overestimate; the result is highly sensitive to this assumption because five rounds of compounding amplify any per-round drift.

- **Positive transfer in joint training.** The model assumes joint training of the three capabilities transfers positively; the supplied summary of [8] states no finding on transfer across differing label spaces, so this is unmeasured. If transfer is negative, the retrained model could get worse at solving between rounds, violating the constant- $\sigma$  assumption.
- **Independence of solves.** The  $\sigma(P - N_t)$  term treats solves as independent Bernoulli trials over the unsolved pool. Difficulty correlation within the pool would make effective progress slower than the geometric law predicts once the easy residue is exhausted.
- **Exogenous pool.**  $P = 10^4$  is fixed; the coupling to procedural generators of the kind described in [9] is flagged, not modeled. If  $P$  grows with the loop, the trajectory is a lower bound.
- **Verification gate assumed sound.** The Lean checker is assumed to accept only sound proofs; the analysis does not model checker bypasses or axioms added mid-loop.
- **"Useful idea" operationalization.**  $p$  is defined as the fraction of recovered ideas that are correct and relevant, but relevance is judged relative to a solving task whose distribution shifts as the corpus grows; the measurement protocol for  $p$  is itself an open design problem.
- **No cost model.** The analysis counts supervision examples, not compute, human effort, or the reverse-engineering training cost; the break-even in Equation (5) compares yields, not costs, and a full cost-adjusted break-even would raise the threshold above  $1/3$ .

**Open questions.** How  $p$  should be measured (expert judgment versus automated checking of idea consequences) is unresolved; whether idea supervision and solution supervision interfere in joint training is unmeasured; and whether the idea distribution extracted from model-generated proofs drifts from that of human proofs, degrading the idea-prediction capability on the remaining unsolved pool, is unmodeled here.

## 7. Practical Implications and Engineering Pathways

**(a) Who would use the result and what changes.** The user is a team building reasoning models for interactive theorem proving, in the position described by [1] and [2]: holding a seed corpus of machine-checked proofs and a pool of unproved formalized statements, and deciding whether to fund a self-improvement loop or to continue paying experts for frontier proofs. If the result holds, the decision changes as follows: before funding the loop, the team runs the cheap measurement of part (c); if  $\hat{A} > 1$ , the team shifts expert effort from writing full frontier proofs to writing a smaller sample of solutions whose ideas the loop propagates, and the frontier advance rate becomes set by the supply of sample solutions rather than by the model's solve rate. The design parameter that changes is the extraction pipeline's precision target: the analysis says the pipeline must clear  $\hat{p} > 1/(m\hat{r})$ , not an arbitrary quality bar, so engineering effort should be allocated to precision measurement and control first.

**(b) Readiness.** This work is analytical and has not been peer reviewed; against the NASA Technology Readiness Level definitions it sits at TRL 1 to 2: basic principles are stated and the loop's data economics are derived on paper, but no component of the loop has been built and validated in the relevant environment by this analysis. The loop specification and Lean instantiation of [1] and [2] are themselves pre-empirical, since those works state that empirical evaluation remains future work. No readiness above TRL 3 is claimed, and no manufacturing readiness is implied.

**(c) The smallest de-risking experiment.** Take 50 to 100 machine-checked solutions from the seed corpus, run the reverse-engineering capability of [1] on each, and have two independent expert judges mark every recovered idea as correct-and-relevant or not. Measure  $\hat{m}$  (ideas per solution),  $\hat{r}$  (fraction recovered, against a human-annotated idea

list), and  $\hat{p}$  (fraction of recovered ideas judged useful), and compute  $\hat{A} = \hat{m}\hat{r}\hat{p}$ . The outcome  $\hat{A} \leq 1$  refutes the amplification claim at those parameter values;  $\hat{A} > 1$  with margin de-risks the loop's economics at a cost of a few expert-days, with no training run. A second, slightly larger test measures the conditioning value of ideas: solve a held-out set of unsolved problems with and without predicted ideas as conditioning and compare solve rates; equality refutes the loop's mechanism.

**(d) Binding constraints.** In the order they bind: first, the precision-measurement protocol (part c) is itself a bottleneck, since judging relevance of extracted ideas requires expert time and inter-judge agreement is unmeasured; second, the verification gate requires every model-generated solution to pass the Lean checker, so the loop's throughput is capped by checker-integrated solving infrastructure, not by model capability alone; third, compute for joint retraining of all three capabilities each round scales with corpus size, and the cumulative corpus of 42791.26 idea pairs plus 4685.59 solved problems at round five sets the retraining set size; fourth, the assumption of positive transfer across the three capabilities (Section 6) is a data and evaluation constraint: measuring it requires held-out evaluation sets for each capability, which must be constructed before the loop starts. Whether idea-extraction quality degrades as the corpus becomes model-dominated is an open question that no supplied entry answers.

**(e) Order-of-magnitude projection.** As a projection with the assumptions of Section 3.6 and no additional ones: five rounds of the loop on a  $10^4$ -problem pool with a  $10^3$ -problem seed yield 42791.26 clean idea pairs and 3685.59 new machine-checked solutions (computed as  $N_5 - N_0 = 4685.59 - 1000$ ). If the expert cost of the seed was one proof per seed problem, the loop's five-round output is equivalent, in clean idea supervision, to 42.79 seed corpora, at the cost of retraining five times and verifying 3685.59 model-generated proofs. These are yield projections only; compute and human-judging costs are not modeled and would reduce the net benefit.

**(f) Risks and misuse paths.** The direct risk is training-signal poisoning at scale: because the corpus becomes 78.66% model-generated by round five under the stated assumptions, any systematic flaw in the extraction pipeline (for example, ideas that encode shortcuts valid only on already-solved problems) is amplified across rounds and could degrade the solver while each individual solution remains machine-verified and therefore passes the soundness gate. A second risk is evaluation contamination: if the unsolved pool overlaps evaluation benchmarks, the loop optimizes toward the evaluation set. Both risks are mitigated by holding out a fixed evaluation pool and by the periodic expert audit of  $\hat{p}$  described in part (c).

## 8. Conclusion

The hindsight-hierarchies loop of [1], [2] converts supplied solutions into idea supervision at a per-round clean amplification  $A = mrp$ , and this single product determines its viability: the loop beats direct fine-tuning on solutions exactly when  $mrp > 1$ , giving a break-even extraction precision of  $p > 1/3$  at  $m = 5$ ,  $r = 0.6$ . Under illustrative assumptions ( $P = 10^4$ ,  $N_0 = 10^3$ ,  $\sigma = 0.1$ ), five rounds grow the solved corpus to  $N_5 = 4685.59$  problems and accumulate 42791.26 clean idea pairs, a  $42.79\times$  amplification of the seed corpus, while the model-generated corpus fraction reaches 78.66%, making extraction precision, not solve rate, the binding quantity. The analysis is entirely projection: its value is that it converts an untested proposal into a falsifiable one, with the smallest decisive experiment being an expert audit of  $\hat{A} = \hat{m}\hat{r}\hat{p}$  on a few dozen supplied solutions, refutable at a cost of expert-days rather than training runs. If that measurement clears unity with margin, the loop's frontier advance becomes limited by the supply of human-written sample solutions rather than by the model's current solve rate, which is the specific design change the analysis was undertaken to enable.

## Provenance and AI Involvement

---

- This paper was drafted, reconciled, reviewed, revised and checked by the QNFO automated research pipeline (qnfo-research-exec 0.16.0-revise-retry). Language models were tools in that process; they are not authors. The named author is accountable for the content.
- Models: drafting by openai/gpt-oss-120b, zai-org/glm-5.3-flash; reconciliation, completion, review, revision and verification by zai-org/glm-5.3-flash, zai-org/glm-5.3, openai/gpt-oss-120b, with the QNFO model router as a fallback.
- Sources: every reference is drawn from a bibliography retrieved from arXiv and the QNFO corpus before drafting; statements about a cited work are restricted to its retrieved title and abstract.
- Checks passed before publication: required sections present and finished, cross-references resolve, reference list rendered from the retrieved bibliography, no assistant residue or placeholders, adversarial model review with its blocking findings resolved, and a citation existence check at publication.
- Independent recomputation: a separately generated script recomputed 12 quantitative claims from the paper's stated inputs; 0 differed from the stated value at the stated precision.
- Not checked: whether each cited source supports the sentence that cites it beyond its abstract; derivation steps other than the recomputed claims. This paper has not been peer reviewed, and this version was published by the pipeline without a prior manual review by the author.

## References

---

[1] TITLE: arXiv Query: `searchquery=&idlist=2610.12168&start=0&max_results=1` [2] Learning to Plan by Looking Back: Hindsight Hierarchies for Training Reasoning Models. arXiv:2610.12168v1. <https://arxiv.org/abs/2610.12168v1> [3] Hierarchical Reinforcement Learning with Hindsight. arXiv:1805.08180v2. <https://arxiv.org/abs/1805.08180v2> [4] Navigating the State of Cognitive Flow: Context-Aware AI Interventions for Effective Reasoning Support. arXiv:2504.16021v1. <https://arxiv.org/abs/2504.16021v1> [5] Reasoning with Justifiable Exceptions in Contextual Hierarchies (Appendix). arXiv:1808.01874v1. <https://arxiv.org/abs/1808.01874v1> [6] Why & When Deep Learning Works: Looking Inside Deep Learnings. arXiv:1705.03921v1. <https://arxiv.org/abs/1705.03921v1> [7] Learning from the Hindsight Plan -- Episodic MPC Improvement. arXiv:1609.09001v2. <https://arxiv.org/abs/1609.09001v2> [8] Transfer Learning with Pre-trained Conditional Generative Models. arXiv:2204.12833v3. <https://arxiv.org/abs/2204.12833v3> [9] Reasoning Core: A Scalable Procedural Data Generation Suite for Symbolic Pre-training and Post-Training. arXiv:2603.02208v1. <https://arxiv.org/abs/2603.02208v1> [10] QNFO: Syntactic Generation [11] QNFO: Impact of Cognitive Linearity on Epistemic Modeling [12] QNFO: Projective Geometric Frameworks for Semantic Structures [13] QNFO: Structured Cross-Domain Translation as a Testable Consilience Mechanism: Protocol Design, Truth-Preservation Conditions, and a Computational Evaluation Blueprint