ISM: Self-Improving Strategy Memory for Continual Mathematical Reasoning

Prakhar Dixit [1] **Tim Oates** [1]

Abstract

We propose **Intelligent Schema Memory (ISM)**, a self-evolving memory-augmented system that improves mathematical reasoning for a frozen LLM under continual learning with hard episodic resets. ISM maintains a compact, self-refined bank of strategy schemas learned from both successful and failed episodes, with symbolic tools that check intermediate steps and certify answers. Without updating model parameters, ISM outperforms passive, retrieval, and reflection baselines on MATH-Hard and OlympiadBench, using 64% and 86% fewer schemas respectively than the strongest passive baseline. These results show that small, actively maintained, and verified strategy memories can support reliable continual mathematical reasoning under strict episodic isolation. The codebase is available at https://github.com/pdx97/ISM.

1. Introduction

Large language models (LLMs) (Zhao et al., 2026) excel at solving isolated mathematical problems, but they frequently stumble when faced with a sequential stream of problems spanning shifting domains. Because each new problem starts from the same frozen parameters, any insights the model gained from a previous task are lost before the next one begins. This highlights the core challenge of continual learning under a frozen backbone: any cumulative improvement must exist entirely outside the model, within an external repository of knowledge. Mathematical reasoning provides an ideal testbed for studying this dynamic, given its clear structural rules and verifiable correctness. Moreover, engineering a system that improves over time without retraining addresses a fundamental question in AI: *How can a frozen LLM self-evolve at mathematical reasoning while keeping its accumulated knowledge bounded, verifiable, and reliable?*

Existing approaches address aspects of this challenge but leave critical gaps. Reflection-based methods like Reflexion (Shinn et al., 2023) generate verbal feedback following a failure, but this feedback remains free text rather than a reusable, structured strategy. Retrieval-augmented setups simply store raw past examples; they grow without bound and rely entirely on surface-level embedding similarity to determine what to retrieve. Meanwhile, skill-library systems like Voyager (Wang et al., 2023) build executable repositories of learned behaviors, but these libraries only expand. They lack mechanisms to verify new entries, consolidate near-duplicates, or retire obsolete strategies. Consequently, current systems fail to self-improve in a controlled manner; they either forget past lessons, bloat their storage, or accumulate noise.

To address these limitations, we propose **Intelligent Schema Memory (ISM)**, a self-evolving memory framework that retrieves strategies and actively regulates its repository via an autonomous self-improvement loop, all while keeping the underlying LLM frozen. ISM maintains a compact bank of strategy schemas. Each schema is split into two components: a *content section* containing the strategy description, solution template, and heuristics injected into the solver prompt; and a *feature hook* containing structural and embedding-based retrieval signals that adapt online based on usage. This separation decouples what a schema knows from how it is discovered, allowing retrieval accuracy to sharpen with experience without altering the underlying strategies.

The memory bank is maintained by seven self-improvement mechanisms: four govern quality and footprint (audit, merge, prune, and promote/demote), two learn symmetrically from outcomes (*reinforce* strengthens successful paths, while *antipattern* records failures to avoid), and one rehabilitates underperforming schemas before they face pruning. Crucially, every memory update is gated by symbolic verification, preventing the bank from reinforcing faulty generalizations. Together, these mechanisms transform external memory from a passive storage unit into an actively maintained, bounded, and self-correcting asset. The combination

[1] Department of Computer Science, University of Maryland, Baltimore County, Baltimore, Maryland, USA . Correspondence to: Prakhar Dixit <pdixit1@umbc.edu>, Tim Oates <oates@umbc.edu>.

The 3rd *AI for Math Workshop* at the 43rd International Conference on Machine Learning (ICML), Seoul, South Korea, 2026.
Copyright 2026 by the author(s).

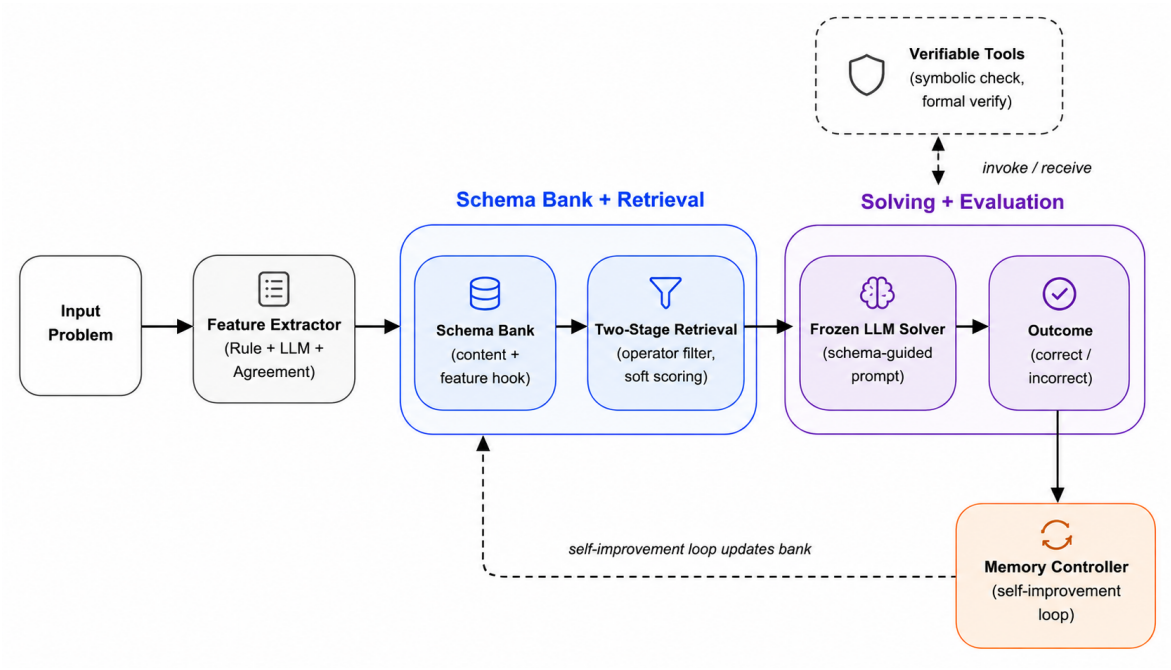

Figure 1. Pipeline of the ISM system. A problem passes through the feature extractor (rule-based and LLM-based with agreement scoring), then the schema bank with two-stage retrieval (operator filter and soft scoring). The retrieved schema's content is injected into the frozen LLM solver, which may invoke verifiable tools. After each episode, the Memory Controller's self-improvement loop (dashed arrow) updates the schema bank.

of symmetric success/failure learning, online-adapting retrieval, lifecycle management, and verification gating is what distinguishes ISM from prior memory-augmented systems built around frozen LLMs.

We evaluate ISM over a 300-episode continual learning stream (Wang et al., 2024) on two competition-level benchmarks, MATH-Hard (Hendrycks et al., 2021) and OlympiadBench (He et al., 2024), against five baselines: vanilla, retrieval, reflection, static schema, and passive schema memory. ISM achieves the highest accuracy on both benchmarks while storing 64% and 86% fewer schemas than the strongest passive baseline, and up to 23× fewer than retrieval-based systems. This performance gap widens as the stream progresses: ISM exhibits greater resilience to domain shifts than any baseline, and its memory bank remains strictly bounded while retrieval-based storage grows linearly with the number of problems encountered.**Our main contributions are**:

- We introduce **ISM**, a self-evolving memory architecture with an autonomous self-improvement loop that retrieves strategies, learns symmetrically from successes and failures, and gates every update through symbolic verification.
- We propose a dual-representation schema design that decouples content (what to do) from retrieval features (when to apply it), letting retrieval adapt online while strategy content stays stable.
- We evaluate ISM under a continual learning protocol with hard episodic resets on MATH-Hard and OlympiadBench, where it outperforms five baselines while using up to 23× less storage.

2. Related Work

In-model memory and continual learning. A recent line of work addresses continual learning by redesigning the model's internal memory substrate. Titans (Behrouz et al., 2024) introduces a neural long-term memory module that learns to memorize at test time, partitioning representation between a short-term attention buffer and a long-term recurrent memory whose parameters update during inference. Nested Learning (Behrouz et al., 2025) generalizes this perspective, recasting a model as a hierarchy of nested optimization problems with distinct update frequencies, fast inner loops for transient context and slow outer loops for stable knowledge, instantiated in the HOPE architecture and its Continuum Memory System. These approaches mitigate catastrophic forgetting by enriching the model's own memory substrate at the architecture level. ISM is complementary: rather than redesign the substrate, we keep the LLM entirely frozen and externalize the memory hierarchy as a symbolic schema bank with explicit lifecycle manage-

ment. The two directions are largely orthogonal and could in principle be combined.

External memory and skill libraries. Several systems attach external memory to frozen models. MemGPT (Packer et al., 2023) introduces tiered context windows backed by a searchable archive, and agentic memory systems organize past episodes through dynamic structures. Skill-library systems such as Voyager (Wang et al., 2023) accumulate executable skills as an agent operates in an environment, appending new skills on success. Retrieval-augmented generation (Lewis et al., 2020) retrieves raw past examples by embedding similarity. ISM differs along three axes: the bank stores compact symbolic schemas rather than executable code or raw examples, retrieval features and bank contents evolve through explicit self-maintenance mechanisms rather than monotonic growth, and every update is gated by symbolic verification before being committed.

Self-reflection and self-improvement. Reflection-based methods generate verbal critiques after failures. Reflexion (Shinn et al., 2023) stores per-episode reflections in natural language, and Self-Refine (Madaan et al., 2023) iterates critique-and-revise within a single problem. STaR (Zelikman et al., 2022) bootstraps reasoning by fine-tuning on self-generated rationales. Most prior systems are asymmetric in their learning signal, accumulating either positive examples (STaR, Voyager) or negative feedback (Reflexion) but not both as structured, retrievable knowledge. ISM treats both as complementary signals through symmetric mechanisms: Self-Reinforce distills positive heuristics from successful episodes, while Self-Antipattern records mistakes to avoid from failures, with both injected into the solver prompt.

Mathematical reasoning and verification. Chain-of-thought prompting (Wei et al., 2022) and program-aided variants (Chen et al., 2022; Gao et al., 2023) elicit step-by-step reasoning from frozen LLMs. Process reward models (Lightman et al., 2023) score reasoning at the step level for verifier-guided generation. The MATH benchmark (Hendrycks et al., 2021) and competition-level evaluations such as OlympiadBench (He et al., 2024) have driven progress in this space. ISM combines schema-guided prompting with formal symbolic verification of intermediate steps and final answers, using verification not only on outputs but as a quality gate on what enters memory.

3. Approach

ISM consists of five components that interact during each episode: (i) a **feature extractor** that converts a problem into a structured representation, (ii) a **schema bank** where each schema is split into content and a feature hook, (iii) a **two-stage retrieval** procedure that selects the most relevant schema for the current problem, (iv) a **frozen LLM solver** that is prompted with the retrieved schema and may optionally invoke verifiable tools, and (v) a **memory controller** that runs a self-improvement loop over the bank after each episode. Figure 1 shows how these components interact. The remainder of this section formalizes the setting, describes each component, and details the verification-gated updates that distinguish ISM from passive memory baselines.

3.1. Problem Setup

We study continual mathematical reasoning under a streaming episode protocol. A problem stream $\mathcal{S} = \{(x_1, y_1), \ldots, (x_T, y_T)\}$ is partitioned into B blocks of N problems each, where each block is drawn from a single mathematical domain (e.g., Algebra, Geometry, Number Theory). At episode t, the system observes problem x_t and produces a candidate answer $\hat{y}_t$. After every block, the domain shifts.

Crucially, each episode is processed under *hard episodic resets*: the model receives no context from previous episodes other than what is explicitly stored in the external memory $\mathcal{M}_t$. Formally, the solver computes

$$\hat{y}_t = f_\theta(x_t, \mathcal{M}_t), \tag{1}$$

where f_θ is a frozen LLM with fixed parameters θ, and $\mathcal{M}_t$ is the schema bank at the start of episode t. The bank is updated to $\mathcal{M}_{t+1}$ by the memory controller after observing the outcome. The frozen parameter constraint isolates the contribution of the external memory from any in-model adaptation.

We evaluate continual learning behavior with four standard metrics: average accuracy, plasticity (accuracy on the current block's domain), stability (accuracy on previously seen domains), forgetting (drop in accuracy on past domains relative to when they were last seen), and backward transfer (BWT, change in accuracy on past domains as new domains are added).

3.2. Feature Extractor

Each problem x_t is mapped to a structured PROBLEMFEATURES object that captures four attributes: an *operator type* (algebraic, number theory, geometric, combinatoric, probability, or calculus), a *structural pattern* (e.g., `two_agents_combined`, `optimization`, `evaluate_expression`), a set of *heuristic signatures* (e.g., `decompose`, `work_backwards`, `apply_theorems`), and a *semantic embedding* of the problem text.

ISM uses a hybrid extractor that combines a rule-based branch (keyword matchers and surface-form patterns) with

an LLM-based branch (a lightweight classification call). The two outputs are reconciled by an agreement score computed as a weighted sum over four attribute-level comparisons:

$$\begin{aligned}\text{conf} = 0.40\,\mathbf{1}[o_r{=}o_l] + 0.30\,\mathbf{1}[p_r \in P_l^{\text{top-2}}] \\ + 0.15\,J(H_r, H_l) + 0.15\,\text{sim}_q(q_r, q_l),\end{aligned} \tag{2}$$

where subscripts r and l denote the rule-based and LLM-based extractions respectively, o is the operator type, p the structural pattern, H the heuristic set, and q the quantity signature. J denotes Jaccard similarity over heuristic sets, and sim_q scores the quantity signature on attribute-level overlap (counts of known and unknown quantities, presence of rates, times, and constraints). The structural match is permissive: p_r counts as matching if it appears in the LLM's top-2 predicted structures $P_l^{\text{top-2}}$. The four weights sum to one, so the threshold $\text{conf} \geq 0.60$ requires the LLM extraction to match at least 60% of the weighted attribute mass with the rule extractor. When this threshold is met, the LLM features are used; otherwise the rule-based features serve as a fallback. The output is a `ProblemFeatures` tuple $(o_t, p_t, H_t, q_t, e_t)$, where the embedding e_t is computed independently by a sentence encoder, that is then passed to the schema bank.

3.3. Two-Stage Retrieval

Given problem features F_t, retrieval selects a schema in two stages.

Stage 1: Operator Filter. A hard filter restricts candidates to schemas whose hook `operator_type` matches F_t.operator. This eliminates entire families of irrelevant schemas (e.g., geometry schemas for a number-theory problem) before scoring.

Stage 2: Soft Scoring. Each surviving candidate s receives a weighted score combining structural, heuristic, quantity, embedding, and prior terms:

$$\text{score}(s, F_t) = \sum_{k \in K} w_k \cdot \text{sim}_k(s, F_t), \tag{3}$$

where $K = \{\text{structural}, \text{heuristic}, \text{quantity}, \text{embedding}, \text{prior}\}$ with weights $w = (0.15, 0.15, 0.05, 0.55, 0.10)$. The embedding similarity dominates because it generalizes across surface forms; structural and heuristic terms add complementary signal. The prior term incorporates the schema's success rate, encouraging the system to retrieve schemas with a track record.

Based on the top score, the retrieval falls into one of three buckets: *high* (score ≥ 0.80, used directly), *medium* (score ≥ 0.45, used with a fallback generic schema attached), or *generic* (score < 0.45, a domain-level generic schema is used and the outcome may trigger Conditional Schema Evolution).

3.4. Frozen LLM Solver and Tools

The retrieved schema's content is injected into a prompt template and passed to a frozen LLM solver via a `hard_reset_call`: a fresh call with no memory of previous episodes. The solver receives the problem text, the schema's description, solution template, and heuristics, and produces an answer.

For challenging problems, the solver can optionally enter an *agentic mode* in which it invokes verifiable tools: a symbolic computation tool that evaluates expressions, a formal answer verifier built on symbolic algebra, a schema lookup tool that surfaces additional candidates, and a failure-search tool that retrieves antipatterns from the replay buffer. Tool use is bounded by a maximum-turn budget and only fires when the initial answer fails an internal sanity check. The agentic mode is gated rather than always-on, keeping inference cost low on routine problems.

3.5. Memory Controller and Self-Improvement Loop

After each episode, the memory controller performs two parallel tracks of updates.

Per-episode updates. The hook of the retrieved schema is updated after each episode via outcome-dependent exponential moving averages on the embedding centroid and success rate (full equations in Appendix A.1).

Periodic self-improvement mechanisms. After a warmup of 10 episodes, seven mechanisms run on independent schedules to maintain the bank:

- **Self-Audit** (every 10 eps): scores each schema as *strong*, *neutral*, *weak*, or *unused* based on success rate and usage count.
- **Self-Correct** (triggered by audit): when a schema is weak with at least 5 uses, an LLM rewrites its content and the success rate is reset to 0.5. After three failed corrections, the schema is escalated to pruning.
- **Self-Merge** (every 20 eps): pairs of schemas whose embedding centroids satisfy $\cos(\mu_i, \mu_j) > 0.88$ are merged via a weighted-centroid combination, with seed schemas protected.
- **Self-Promote / Demote** (triggered by audit): schemas with $\rho_s \geq 0.80$ have their retrieval scores boosted by 2%; schemas with $\rho_s \leq 0.40$ are penalized by 5%.
- **Self-Prune** (every 25 eps): non-seed schemas with

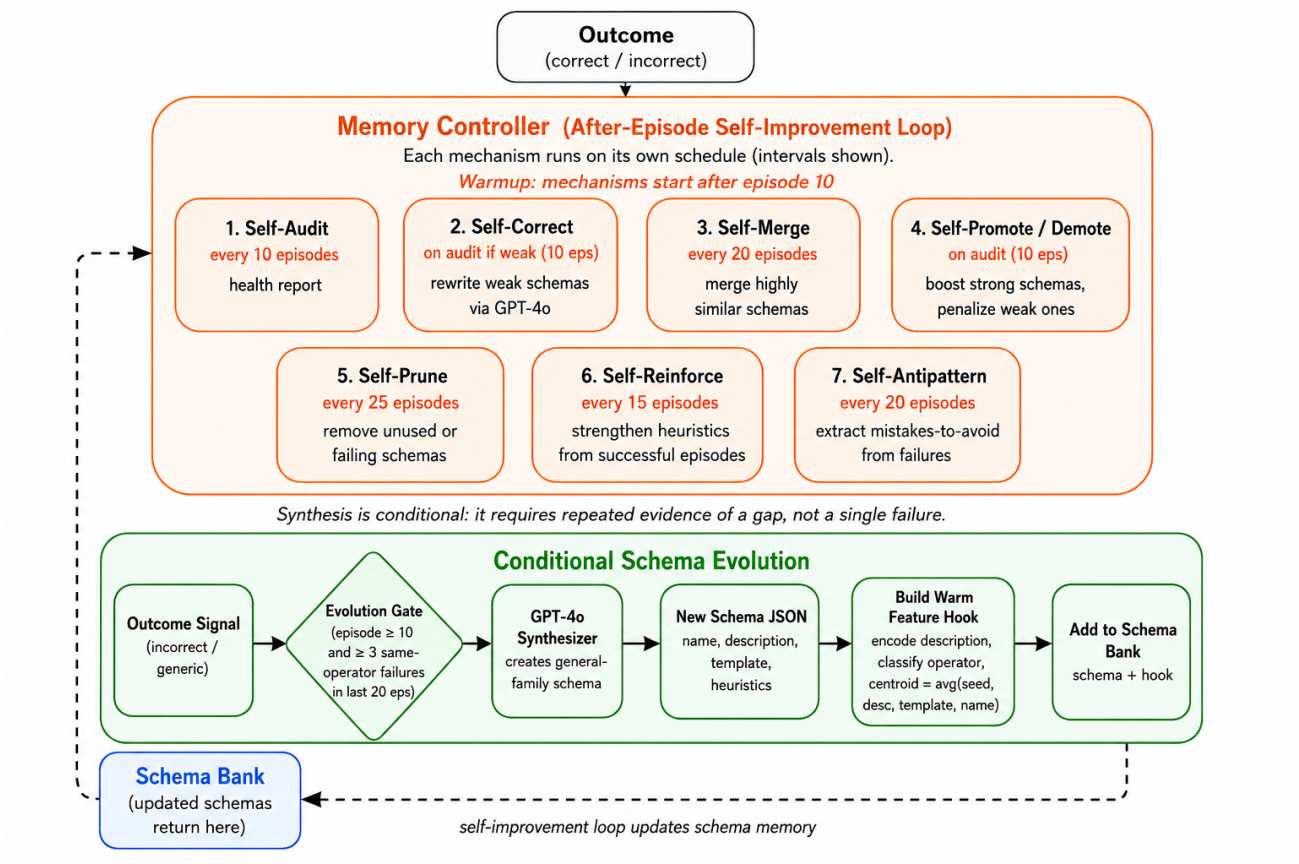

Figure 2. Memory Controller and conditional schema evolution. After each episode, the outcome is passed to the Memory Controller, which runs seven self-improvement mechanisms on independent schedules (top, orange): Self-Audit produces a health report every 10 episodes; Self-Correct rewrites weak schemas flagged by the audit; Self-Merge consolidates near-duplicates every 20 episodes; Self-Promote/Demote reweights schemas based on success rate; Self-Prune removes unused or failing schemas every 25 episodes; Self-Reinforce strengthens heuristics from successful episodes every 15; and Self-Antipattern extracts mistakes-to-avoid from failures every 20. Schema synthesis (bottom, green) is conditional: the Evolution Gate fires only when at least 10 episodes have elapsed and at least three same-operator failures have occurred in the last 20, after which a synthesizer creates a new general-family schema with a warm-started feature hook. The dashed arrow closes the self-improvement loop back to the Schema Bank.

zero uses or persistent failure are removed from the bank.

- **Self-Reinforce** (every 15 eps): for strong schemas, an LLM distills new positive heuristics from recent successful episodes and appends them to the schema content.
- **Self-Antipattern** (every 20 eps): for schemas associated with at least three failures, an LLM extracts mistakes-to-avoid from the failed episodes and records them in the content.

Conditional schema evolution. On incorrect or generic outcomes, the system evaluates an *evolution gate*: if the current episode is at least 10 and at least three same-operator failures have occurred in the last 20 episodes, the synthesizer is invoked. The synthesizer is an LLM call that produces a new general-family schema in structured JSON form, after which a warm feature hook is built (operator classified, centroid initialized as the average of seed, description, template, and name embeddings) and the schema is added to the bank.

Verification gate. Every memory update, including hook adjustment, merge, content rewrite, promote/demote, reinforce, antipattern, and synthesis, is gated by a symbolic verification step. For example, before Self-Reinforce appends new heuristics, the underlying solution must have been verified by symbolic computation, and before Self-Merge consolidates two schemas, their shared use cases must symbolically check out. This gate prevents the bank from reinforcing incorrect generalizations, which is the failure mode of unverified memory systems. For more additional details about the Schema Bank refer to Appendix A.1

4. Experiments

4.1. Benchmarks

We evaluate on two competition-level mathematical reasoning benchmarks of different difficulty. **MATH-Hard** consists of the level-5 (hardest) problems from the MATH dataset (Hendrycks et al., 2021), drawn from Algebra, Geometry, and Number Theory. These are high-school competition problems with short numerical or symbolic answers, hard enough to make a frozen LLM stumble often without good prompting. **OlympiadBench** (He et al., 2024) is a collection of olympiad-level mathematics problems, taken from international and national competitions, that are noticeably harder than MATH-Hard: longer setups, multi-step reasoning, and answers that often require careful case analysis or symbolic manipulation. For both benchmarks we run a 300-episode continual learning stream split into blocks by domain, with hard shifts between blocks. Every episode is processed under a hard episodic reset, so nothing carries over from one problem to the next except what the system has explicitly stored in its memory bank.

4.2. Baselines

We compare ISM against five baselines that cover the main approach families: (1) **Vanilla LLM**, the frozen model with no external memory; (2) **Static Schema**, one fixed prompt template used for every problem, with no retrieval and no updates; (3) **RAG-over-Examples**, where every past episode is kept as an exemplar and the closest one by embedding similarity is fetched at inference time; (4) **Reflexion** (Shinn et al., 2023), which adds a short verbal reflection to a growing buffer after each episode; and (5) **Passive Schema Memory**, where schemas are created and stored just like in ISM but the seven self-improvement mechanisms are turned off. The gap between Passive and ISM tells us what active maintenance actually buys.

4.3. Metrics

We report final accuracy averaged over all 300 episodes along with four continual learning metrics. *Plasticity* is the accuracy on the first 10 episodes of each new domain block, showing how quickly the system picks up a new domain. *Stability* is the accuracy on the last 10 episodes of each domain, showing how well past performance is held on to. *Forgetting* is how much accuracy drops on a domain compared to when it was last seen. *Backward transfer* (BWT) tracks whether past domains get better or worse as new ones come in; a positive number means past domains keep improving. We also report the final bank size as a simple measure of storage cost.

4.4. Implementation

We use `gpt-4.1-mini` as the frozen solver across all systems, called via a hard-reset API call with no message history and temperature 0.0. Self-improvement mechanisms that require an LLM (Self-Correct, Self-Merge, Self-Reinforce, Self-Antipattern, and schema synthesis) use `gpt-4o`, and problem embeddings come from `text-embedding-3-small`.Retrieval works in two stages. Stage 1 hard-filters schemas by operator type. Stage 2 ranks the remaining candidates by a weighted score over structural, heuristic, quantity, embedding, and prior terms with weights $(0.15, 0.15, 0.05, 0.55, 0.10)$. Retrievals fall into three buckets by top score: high (≥ 0.80, used directly), medium (≥ 0.45, used with a generic fallback), and generic (< 0.45).The hook of the retrieved schema is updated after each episode with outcome-dependent EMAs. The embedding centroid moves with $\alpha_e = 0.04$ on a success and 0.01 on a failure, and the success rate moves with $\alpha_r = 0.15$ on a success and 0.10 on a failure.After a warmup of 10 episodes, the seven self-improvement mechanisms run on independent schedules: Self-Audit every 10 episodes (which can trigger Self-Correct and Self-Promote/Demote), Self-Reinforce every 15, Self-Merge and Self-Antipattern every 20, and Self-Prune every 25. The evolution gate fires only when at least 10 episodes have elapsed and at least three same-operator failures have occurred in the last 20 episodes, so a new schema is synthesised only when the gap is real.

4.5. Quantitative Results

As seen in Table 1, the results on both benchmarks are brought together. ISM comes out ahead on accuracy, stability, forgetting, and backward transfer on both MATH-Hard and OlympiadBench, while keeping a much smaller bank than any retrieval baseline.

Accuracy and continual learning metrics. ISM reaches 80.67% on MATH-Hard and 61.67% on OlympiadBench, beating the strongest baseline (Passive Schema Memory) by exactly 2.00 points on each. The cumulative accuracy curves in Figures 7 and 8 show ISM pulling ahead in the early episodes and holding the lead through every domain shift. Stability goes from 82.00% to 84.67% on MATH-Hard and from 62.00% to 64.00% on OlympiadBench, while forgetting drops from 0.10 to 0.07 and from 0.10 to 0.03 respectively, both compared to Passive. On OlympiadBench, ISM hits backward transfer of $+0.03$, meaning past domains actually keep getting better as new ones arrive. Reflexion is the only other system with a positive BWT on OlympiadBench ($+0.01$), but it sits at much lower accuracy overall.

Memory efficiency. ISM keeps 17 schemas to Passive's 47 on MATH-Hard (64% fewer) and 13 to Passive's 91 on

Table 1. Main results on MATH-Hard and OlympiadBench over a 300-episode continual learning stream. Best values per benchmark in bold. ISM is best on accuracy, stability, and forgetting on both benchmarks, with positive backward transfer on OlympiadBench. RAG and Reflexion store one entry per episode, capped at the stream length.

System	Acc.	Plast.	Stab.	Forget.	BWT	Bank
MATH-Hard						
Vanilla LLM	48.00	46.00	50.00	0.10	−0.08	0
RAG-over-Examples	57.00	54.00	60.00	0.12	−0.12	300
Reflexion	55.67	52.67	58.67	0.13	−0.13	300
Static Schema	78.67	76.00	81.33	0.08	−0.06	1
Passive Schema Memory	78.67	75.33	82.00	0.10	−0.07	47
ISM (ours)	**80.67**	**76.67**	**84.67**	**0.07**	−0.05	**17**
OlympiadBench						
Vanilla LLM	23.67	24.00	23.33	0.05	−0.03	0
Reflexion	29.67	26.67	32.67	0.02	+0.01	300
RAG-over-Examples	33.33	32.67	34.00	0.02	−0.02	300
Static Schema	59.67	58.00	61.33	0.06	−0.00	1
Passive Schema Memory	59.67	57.33	62.00	0.10	−0.02	91
ISM (ours)	**61.67**	**59.33**	**64.00**	**0.03**	+0.03	**13**

OlympiadBench (86% fewer). The retrieval baselines hit 300 entries on both benchmarks. So ISM uses 17.6× fewer entries than RAG and Reflexion on MATH-Hard and 23× fewer on OlympiadBench. The gap grows with difficulty: harder problems push Passive to synthesize more schemas, but the merge and prune mechanisms in ISM keep the bank compact either way.

Figures 9 and 10 show how the bank changes over time. On MATH-Hard, ISM's bank climbs during the early Geometry block (episodes 100–145) and then *shrinks* from 26 down to 15 as Self-Prune and Self-Merge clean it up. Passive grows steadily from 6 to 47 with nothing pulling it back. On OlympiadBench, the gap is even bigger: ISM stays in the 10–22 range the whole stream, while Passive climbs linearly to 91. The repeated dips in the ISM curve are the prune and merge cycles doing their job; without them, the bank would look like the Passive curve.

4.6. Qualitative Results

To see where active memory actually helps, we picked out episodes where ISM got the right answer while three or more baselines got it wrong.

Each of these schemas was synthesized after the evolution gate caught a repeated pattern of same-operator failures. After that, they were reused 5, 12, and 19 times respectively, with success rates between 0.63 and 0.76. When the same kind of pattern shows up again later in the stream, ISM already has something ready to apply. On Episode 235 the Passive baseline returned 121 (wrong) while ISM returned 93 (right): both had a relative-primality schema in their bank, but only ISM had spent the episodes in between sharpening its feature hook through self-improvement to actually retrieve it when it mattered.For additional quantitative results and qualitative case studies, refer to Appendix C.

5. Limitations

This work presents an initial study of Intelligent Schema Memory (ISM) under continual mathematical reasoning with frozen LLMs, and several limitations remain.

First, all experiments are conducted using a single random seed and a single ordering of the continual learning stream. Continual learning systems are known to be sensitive to both initialization and stream composition, since different domain orders alter which transitions stress adaptation and forgetting. As a result, the reported gains should be interpreted as preliminary evidence rather than definitive estimates of performance. Multi-seed evaluation, shuffled stream orderings, and variance reporting are important next steps that we leave for future work.

Second, we do not include per-mechanism ablations isolating the contribution of individual components such as Self-Reinforce, Self-Antipattern, Self-Merge, Self-Prune, the verification gate, or conditional schema synthesis. Instead, we evaluate an aggregate ablation through the Passive Schema Memory baseline, where all self-improvement mechanisms are disabled simultaneously. While this comparison demonstrates the overall value of active memory maintenance, it does not clarify which mechanisms contribute most strongly to the observed improvements or which may be redundant.

Third, our baselines focus primarily on memory-oriented approaches, including retrieval-based storage, reflection buffers, and passive schema memory. We do not compare against search-heavy reasoning systems such as Tree-of-Thoughts (Yao et al., 2023), graph-structured memory architectures (Oliveira et al., 2025), or fully agentic ReAct-

style frameworks (Yao et al., 2022). These systems target somewhat different aspects of reasoning and control, and a broader comparison across reasoning paradigms would strengthen the evaluation.

In addition, all experiments are performed using a single frozen LLM backbone. Although ISM is designed to be model-agnostic, we have not yet evaluated whether the same trends hold consistently across smaller or larger models. The evaluation is also limited to two competition-level mathematical reasoning benchmarks. Whether the same lifecycle dynamics and symmetric success/failure learning mechanisms generalize to other domains, such as formal proof generation, code reasoning, or broader scientific problem solving, remains an open question.

Finally, the verification gate currently relies on symbolic computation tools that are well suited for numerical and short symbolic answers, but less applicable to proof-oriented tasks where correctness itself may require complex reasoning. Extending verification to richer forms of mathematical reasoning is an important direction for future work. Similarly, the agentic tool-use loop in ISM is selectively activated only after failed initial attempts, and we do not evaluate settings where tool use is persistent or always-on.

6. Conclusion

We presented **Intelligent Schema Memory (ISM)**, a self-evolving memory system that lets a frozen LLM improve at mathematical reasoning under hard episodic resets. ISM keeps a compact bank of strategy schemas, each split into content that goes into the solver prompt and a feature hook that adapts online, with a self-improvement loop that audits, merges, prunes, reinforces, and records antipatterns, all gated by symbolic verification. Across MATH-Hard and OlympiadBench, ISM achieves the strongest accuracy, stability, and forgetting metrics among the systems we tested, while keeping a memory bank 64 to 86 percent smaller than the strongest non-active baseline and up to 23 times smaller than retrieval-based systems.

These results suggest that small, actively maintained, verifiable strategy memory is a viable path for continual reasoning in frozen-LLM systems. The next steps are multi-seed evaluation, per-mechanism ablations, and extending ISM beyond mathematical reasoning to domains where verification is less direct.

References

A. Behrouz, P. Zhong, and V. Mirrokni. Titans: Learning to memorize at test time. *arXiv preprint arXiv:2501.00663*, 2024.

A. Behrouz et al. Nested learning: The illusion of deep learning architectures. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

W. Chen et al. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *arXiv preprint arXiv:2211.12588*, 2022.

L. Gao et al. Pal: Program-aided language models. 2023.

C. He, R. Luo, Y. Bai, S. Hu, Z. Thai, J. Shen, J. Hu, X. Han, Y. Huang, Y. Zhang, et al. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3828–3850, 2024.

D. Hendrycks et al. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

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

H. Lightman et al. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

A. Madaan, N. Tandon, P. Gupta, S. Hallinan, L. Gao, S. Wiegreffe, U. Alon, N. Dziri, S. Prabhumoye, Y. Yang, S. Gupta, B. P. Majumder, K. Hermann, S. Welleck, A. Yazdanbakhsh, and P. Clark. Self-refine: Iterative refinement with self-feedback, 2023. URL https://arxiv.org/abs/2303.17651.

A. A. Oliveira, M. Espadoto, R. M. Cesar Jr, and R. Hirata Jr. Knowledge graphs as structured memory for embedding spaces: From training clusters to explainable inference. *arXiv preprint arXiv:2511.14961*, 2025.

C. Packer, S. Wooders, K. Lin, V. Fang, S. G. Patil, I. Stoica, and J. E. Gonzalez. Memgpt: Towards llms as operating systems. *arXiv preprint arXiv:2310.08560*, 2023.

N. Shinn, F. Cassano, A. Gopinath, K. Narasimhan, and S. Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in neural information processing systems*, 36:8634–8652, 2023.

G. Wang, Y. Xie, Y. Jiang, A. Mandlekar, C. Xiao, Y. Zhu, L. Fan, and A. Anandkumar. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023.

L. Wang, X. Zhang, H. Su, and J. Zhu. A comprehensive survey of continual learning: Theory, method and application. *IEEE transactions on pattern analysis and machine intelligence*, 46(8):5362–5383, 2024.

J. Wei et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 2022.

S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.

S. Yao, D. Yu, J. Zhao, I. Shafran, T. Griffiths, Y. Cao, and K. Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822, 2023.

E. Zelikman, Y. Wu, J. Mu, and N. D. Goodman. Star: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 2022.

W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong, Y. Du, C. Yang, Y. Chen, Z. Chen, J. Jiang, R. Ren, Y. Li, X. Tang, Z. Liu, P. Liu, J.-Y. Nie, and J.-R. Wen. A survey of large language models, 2026. URL https://arxiv.org/abs/2303.18223.

A. Appendix

A.1. Schema Bank

The schema bank $\mathcal{M}_t$ stores a set of *strategy schemas*. Each schema s is split into two complementary parts:

- **Content section**, containing the schema's name, natural-language description, solution template, and a set of heuristics. The content is what gets injected into the solver's prompt and is largely stable over time.
- **Feature hook**, containing the retrieval-side signals: *operator type*, *structural pattern*, *heuristic signature*, *quantity signature*, an *embedding centroid* (maintained as an exponential moving average over the embeddings of problems the schema has handled), a *success rate* (also EMA), and *usage count*.

The hook adapts online while the content stays fixed unless rewritten by Self-Correct. This separation lets retrieval sharpen with experience without destabilizing the strategies themselves. After each use of schema s on problem x_t with embedding e_t and outcome $r_t \in \{0, 1\}$, the hook updates its centroid and success rate via outcome-dependent EMAs:

$$\mu_s \leftarrow (1 - \alpha_e^{(r_t)})\mu_s + \alpha_e^{(r_t)} e_t, \tag{4}$$

$$\rho_s \leftarrow (1 - \alpha_r^{(r_t)})\rho_s + \alpha_r^{(r_t)} r_t, \tag{5}$$

where the rates are deliberately asymmetric. The centroid uses $\alpha_e^{(\text{correct})} = 0.04$ and $\alpha_e^{(\text{wrong})} = 0.01$: successful retrievals pull the centroid harder toward exemplars the schema handled well, while failed retrievals leave the centroid largely intact. The success rate uses $\alpha_r^{(\text{correct})} = 0.15$ and $\alpha_r^{(\text{wrong})} = 0.10$: positive evidence registers slightly faster than negative evidence, so newly-strong schemas are recognized quickly while occasional misses do not over-penalize them.

Per-domain plasticity and stability. Figure 5 breaks plasticity and stability down by domain on OlympiadBench. ISM matches or beats Passive on plasticity in every domain, with the biggest gain in Geometry (40% vs 30%). On stability, ISM holds its own with the best baselines and pulls ahead in Number Theory. The same picture shows up on MATH-Hard (Figure 6): ISM ties Static and Passive on Algebra, wins on Geometry stability, and slips a bit on Number Theory stability.

Schema Bank Composition The composition of the two banks looks very different. On MATH-Hard, Passive ends with 47 schemas, but 28 of them (60%) were never used again after being created. On OlympiadBench, Passive ends with 91 schemas and 62 of them (68%) were never used again. These are schemas that were synthesized once on a failure and then never retrieved, so they take up space without giving anything back. Figure 11 shows what ISM's bank looks like on MATH-Hard: the seed schemas (Number Theory, Algebra, Geometry, Combinatorics) carry most of the load, but synthesized schemas earn their place too, with *Function Difference Matching* hitting 45 uses at 89% accuracy. Figure 12 shows the same picture on OlympiadBench: every synthesized schema that survived has at least 3 uses and a success rate above 0.59, with *Floor Function Quotient Problems* and *Tangency and Collinearity* above 0.96. Self-Prune cuts off the long tail of unused schemas that Passive holds on to forever.

B. Wins and losses against each baseline.

Figure 3 compares ISM head-to-head with each baseline on OlympiadBench. Against the unmanaged-memory baselines the lead is huge: $+114$ episodes over Vanilla, $+96$ over Reflexion, $+85$ over RAG. Against the schema-based controls the lead is small but consistent: $+6$ over Static and $+6$ over Passive. The MATH-Hard picture (Figure 4) is the same shape: $+98$, $+75$, $+71$ over the unmanaged baselines, and $+6$ each over Static and Passive. The steady $+6$ over the strongest controls is exactly the 2-point gap in the table.

C. Qualitative Case Studies

We present five representative episodes where ISM produced the correct answer through a *synthesised* schema while several baselines failed. These cases illustrate how the evolution gate produces durable, transferable strategies rather than one-shot patches: every schema below was created mid-stream by the synthesiser, accumulated meaningful reuse, and reached a positive success rate.

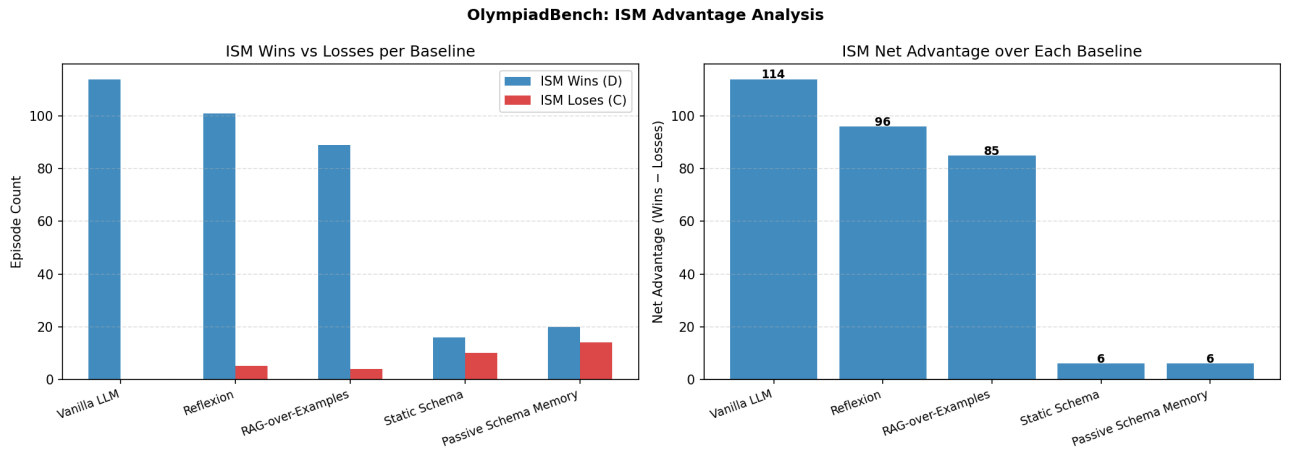

Figure 3. ISM head-to-head against each baseline on OlympiadBench. **Left:** wins (ISM correct, baseline wrong) versus losses (baseline correct, ISM wrong) per baseline. **Right:** net advantage (wins minus losses). ISM dominates the unmanaged-memory baselines ($+114$ over Vanilla, $+96$ over Reflexion, $+85$ over RAG) and holds a steady $+6$ lead over both schema-based controls (Static, Passive). The consistent $+6$ over the strongest baselines aligns with the 2-point accuracy gap

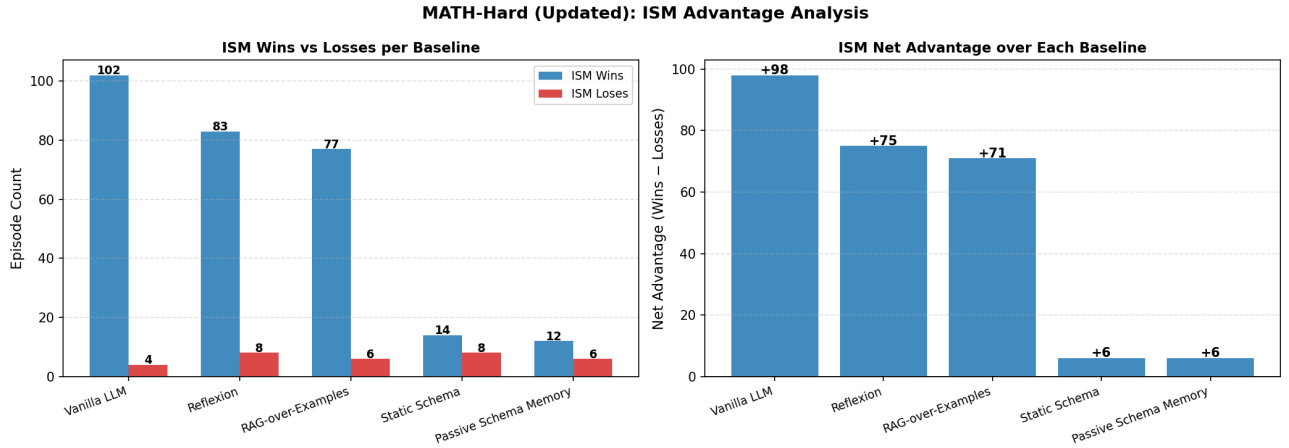

Figure 4. ISM head-to-head against each baseline on MATH-Hard. **Left:** wins versus losses per baseline. **Right:** net advantage. The pattern mirrors OlympiadBench: large net leads over the unmanaged baselines ($+98$ over Vanilla, $+75$ over Reflexion, $+71$ over RAG) and a $+6$ lead over both schema-based controls (Static, Passive).

C.1. OlympiadBench Case Studies

EPISODE 213 (NUMBER THEORY)

Problem.

> *Let p be a prime number. If p years ago, the ages of three children formed a geometric sequence with a sum of p and a common ratio of 2, compute the sum of the children's current ages.*

Schema retrieved. `Simultaneous Polynomial Root Problems` (synthesised; success rate 0.63, used 12 times).

Schema description. *Determine relationships between the roots of multiple polynomial equations with constraints or shared roots.*

Heuristics applied. introduce variable; decompose; use symmetry.

Outcome. ISM produced the correct answer $\boxed{28}$. All four baselines answered incorrectly (e.g., Vanilla LLM predicted $4p$).

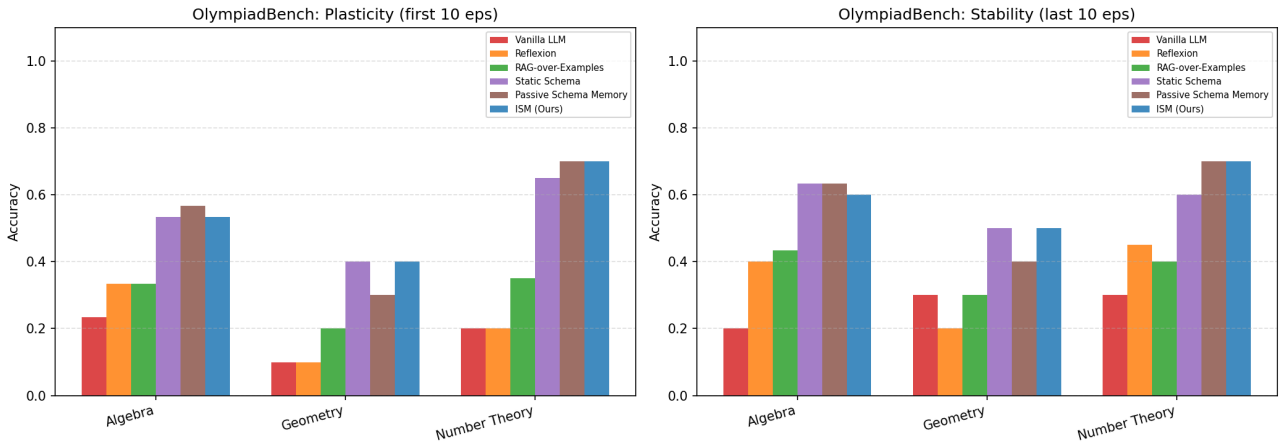

Figure 5. OlympiadBench plasticity (first 10 episodes of each block, left) and stability (last 10 episodes, right) per domain. ISM matches or beats Passive Memory on both metrics across Algebra, Geometry, and Number Theory.

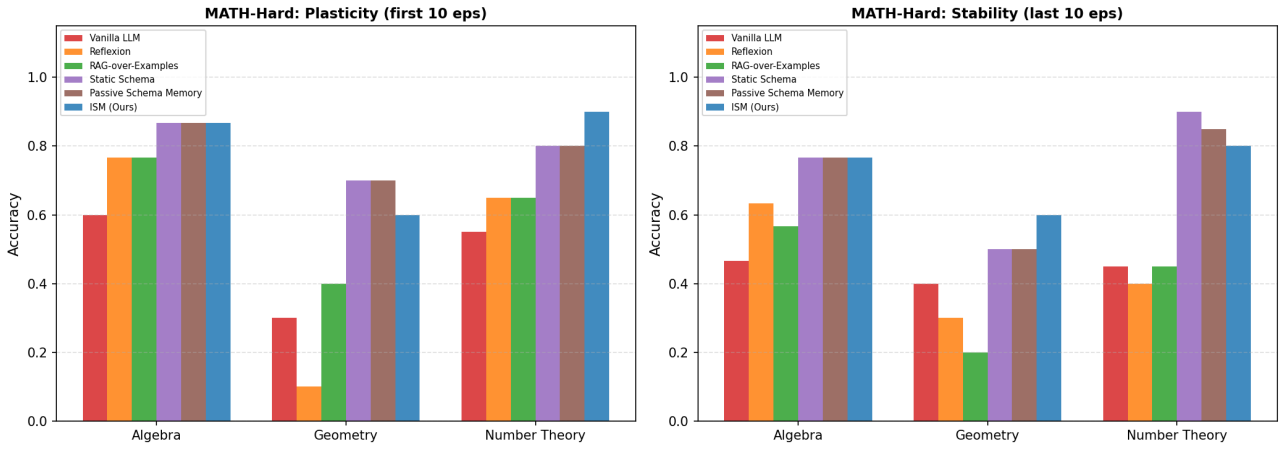

Figure 6. MATH-Hard plasticity (first 10 episodes of each block, left) and stability (last 10 episodes, right) per domain.

EPISODE 235 (NUMBER THEORY)

Problem.

> *Compute the third least positive integer n such that each of n, $n+1$, and $n+2$ is a product of exactly two (not necessarily distinct) primes.*

Schema retrieved. `Relative Primality in Sets` (synthesised; success rate 0.67, used 5 times).

Schema description. *Find the smallest set of integers constructed under specific rules where no element is relatively prime to the product of the others.*

Heuristics applied. decompose; find pattern; introduce variable.

Outcome. ISM produced the correct answer $\boxed{93}$. All four baselines answered incorrectly (e.g., Passive Schema Memory predicted 121).

EPISODE 286 (ALGEBRA)

Problem.

> *The equations $x^3 + Ax + 10 = 0$ and $x^3 + Bx^2 + 50 = 0$ have two roots in common. Compute the product of these common roots.*

Schema retrieved. `Nested Function Zeros` (synthesised; success rate 0.76, used 19 times).

Schema description. *Solve for the unknown polynomials and their compositions when dealing with nested polynomial equations and specific shared-root conditions.*

Heuristics applied. analyse degrees and symmetry; consider boundary or extreme values; use composition properties creatively.

Outcome. ISM produced the correct answer $\boxed{5\sqrt[3]{4}}$. All four baselines failed to produce a valid answer.

C.2. MATH-Hard Case Studies

EPISODE 122 (GEOMETRY)

Problem.

> *In the diagram, two pairs of identical isosceles triangles are cut off of square $ABCD$, leaving rectangle $PQRS$. The total area cut off is $200\ m^2$. What is the length of PR, in meters?*

Schema retrieved. `Composite Geometric Areas` (synthesised; success rate 0.61, used 5 times).

Schema description. *Problems involving the calculation of areas formed by overlapping geometric figures such as squares, circles, and triangles.*

Heuristics applied. decompose; change representation; use symmetry.

Outcome. ISM produced the correct answer $\boxed{20}$. All four baselines answered incorrectly (e.g., Passive Schema Memory predicted $5\sqrt{10}$).

EPISODE 154 (ALGEBRA)

Problem.

> *Let f be defined by*
>
> $$f(x) = \begin{cases} 2 - x & \text{if } x \le 1, \\ 2x - x^2 & \text{if } x > 1. \end{cases}$$
>
> *Calculate $f^{-1}(-3) + f^{-1}(0) + f^{-1}(3)$.*

Schema retrieved. `Function Difference Matching` (synthesised; success rate 0.97, used 85 times).

Schema description. *Problems involving determining a parameter such that the difference between the evaluations of two functions at a target value matches a given constraint.*

Heuristics applied. decompose; introduce variable; work backwards.

Outcome. ISM produced the correct answer $\boxed{4}$. All three baselines failed to produce a valid answer.

C.3. Hyperparameters

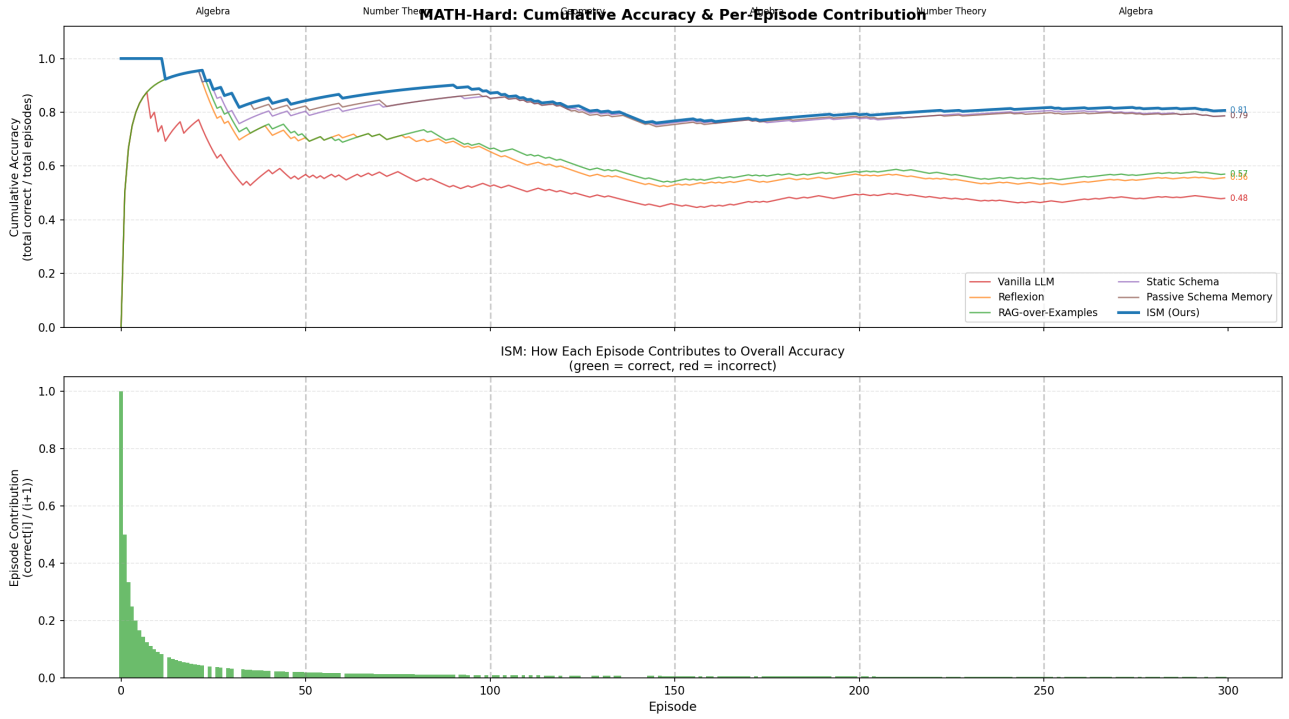

Figure 7. Cumulative accuracy across the 300-episode MATH-Hard stream. ISM leads from early episodes onward and stays ahead through every domain transition (dashed vertical lines), finishing at 0.81. Static Schema and Passive Memory finish at 0.79; RAG and Reflexion stay near 0.56; Vanilla LLM ends at 0.48.

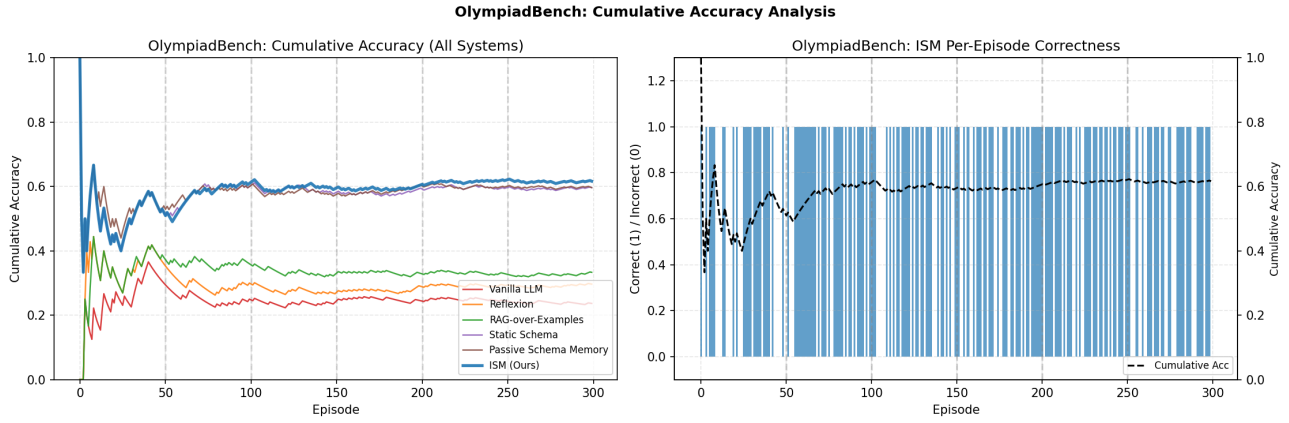

Figure 8. Cumulative accuracy on OlympiadBench (left) and ISM's per-episode correctness with running cumulative accuracy (right). The schema-based systems (ISM, Passive, Static) cluster around 0.60, with ISM slightly above. RAG, Reflexion, and Vanilla stay below 0.35.

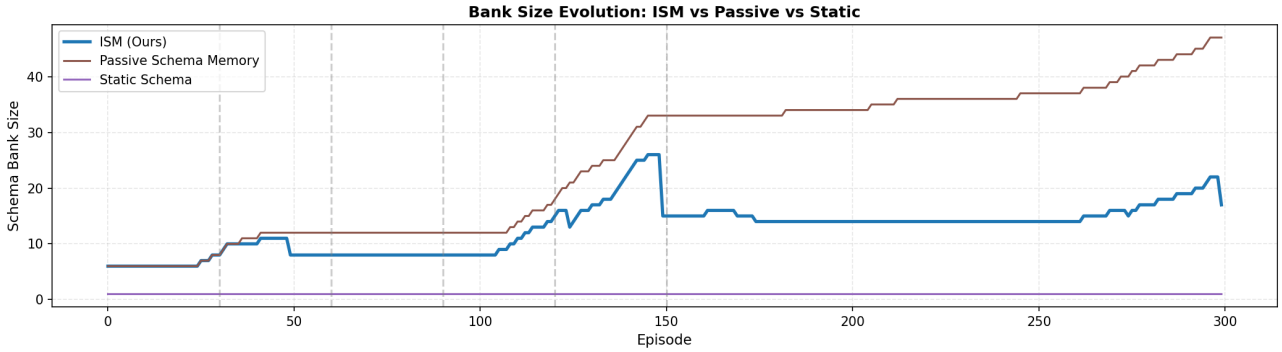

Figure 9. Schema bank size over the MATH-Hard stream. ISM's bank grows during exploration and then drops when Self-Prune and Self-Merge fire (around episode 150 and again past 270). Passive keeps growing.

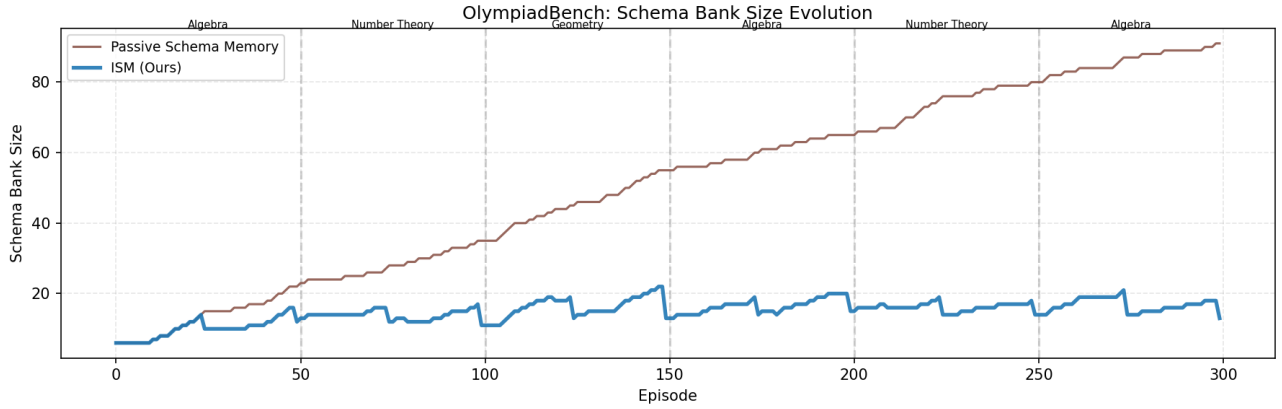

Figure 10. Schema bank size over the OlympiadBench stream. With lifecycle management, the ISM bank stays in the 10–22 range, while Passive climbs to 91. Domain labels at top show which subject each block is drawn from.

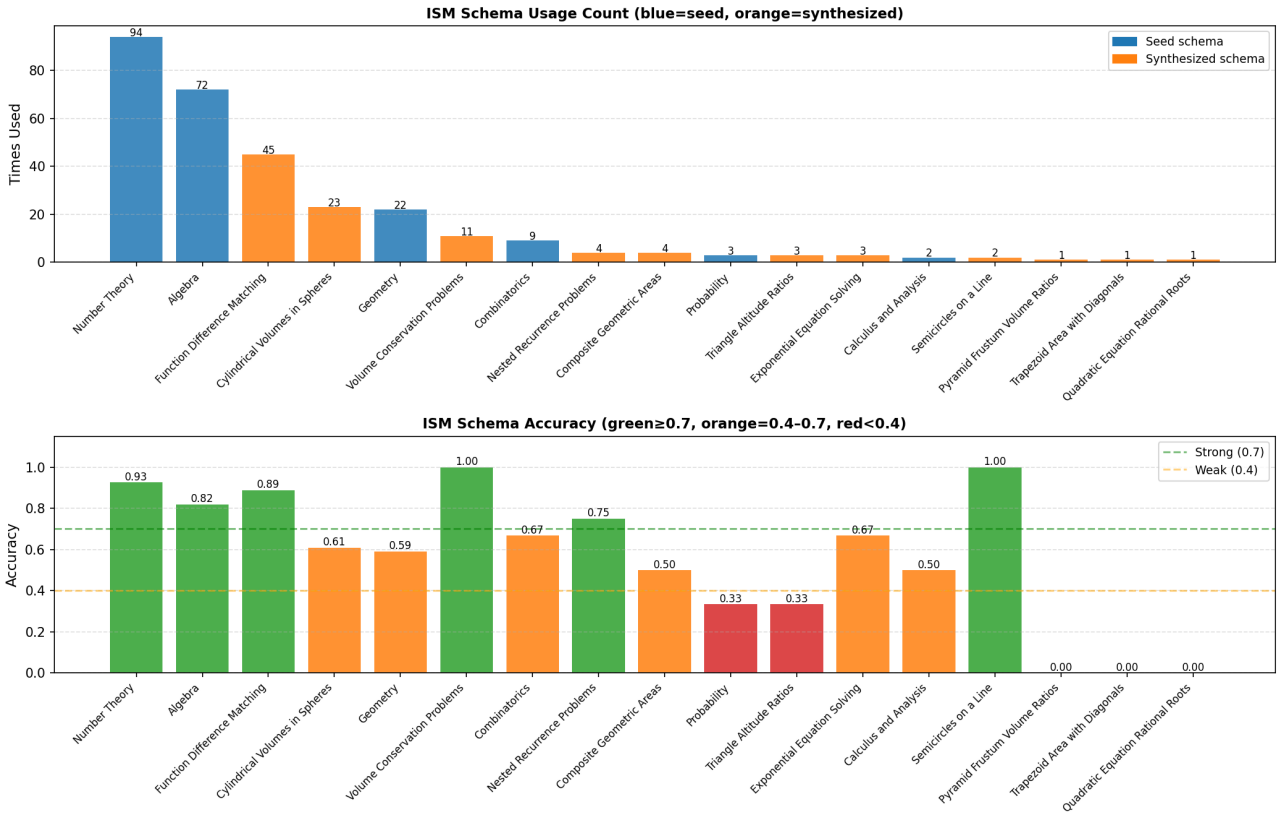

Figure 11. ISM's schema bank on MATH-Hard at the end of the stream. Top: usage count per schema (blue = seed, orange = synthesized). Bottom: success rate per schema. Most retained schemas sit above the strong-schema threshold of 0.7 (green dashed line).

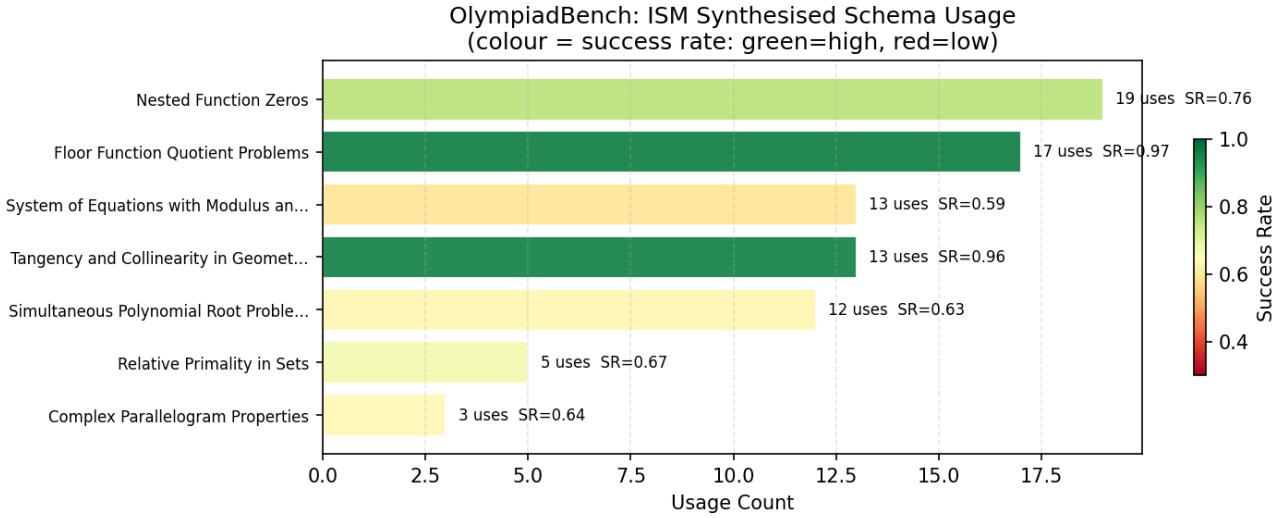

Figure 12. ISM synthesized schemas on OlympiadBench. Every retained synthesized schema has at least three uses and a success rate above 0.59. Color shows success rate (green = high, red = low).

Parameter	Value
Models	
Solver LLM	gpt-4.1-mini
Synthesizer LLM	gpt-4o
Embedding model	text-embedding-3-small
Feature extractor	
Agreement threshold	0.60
Retrieval scoring weights	
Operator filter weight	0.00 (hard filter)
Structural weight	0.15
Heuristic weight	0.15
Quantity weight	0.05
Embedding weight	0.55
Prior (success rate) weight	0.10
Retrieval buckets	
High confidence	≥ 0.80
Medium confidence	≥ 0.45
Generic	< 0.45
EMA rates (centroid)	
Success α_e	0.04
Failure α_e	0.01
EMA rates (success rate)	
Success α_r	0.15
Failure α_r	0.10
Mechanism intervals (episodes)	
Self-Audit	10
Self-Reinforce	15
Self-Merge	20
Self-Antipattern	20
Self-Prune	25
Replay check	10
Mechanism thresholds	
Warmup episodes	10
Self-Correct min uses	5
Self-Correct fail-then-prune	3
Self-Merge cosine threshold	0.88
Self-Promote success rate	≥ 0.80
Self-Promote boost	$+2\%$
Self-Demote success rate	≤ 0.40
Self-Demote penalty	-5%
Evolution gate	
Minimum episode	10
Failures in window	≥ 3 in last 20
Replay buffer	
Max size	500
Failure bias on sampling	70%

Table 2. Full hyperparameter configuration for ISM.