MEMORY THAT IMAGINES: LEARNER MENTAL WORLD MODELS AS MEMORY FOR LONG-HORIZON TUTORING AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Agent memory systems answer what happened. They store, retrieve and summarize interaction logs. A tutoring agent needs its memory to answer two other questions, what the learner knows now and what a candidate teaching action does. This paper casts the memory of a tutoring agent as a mental world model of the learner, a fixed-size belief state. The belief state supports three operations, WRITE (posterior update), READ (mastery with uncertainty) and IMAGINE (closed-loop rollout of candidate teaching sequences). Under a latent-state assumption, the belief retains all the evidence the history holds about future responses, whereas a log cut to any fixed budget can lose that evidence. The memory is a recurrent state-space model trained with closed-loop overshooting to roll forward on its own sampled responses. Two design choices adapt the model to learner data. Each exercise is bound to its response. Prediction and reconstruction use separate heads, because sharing one head costs 6.8 AUC points. On three knowledge-tracing benchmarks, this memory predicts the next 100 responses 8.5–16.2 AUC points better than a retrieval memory or a Bayesian Knowledge Tracing state, and better than two LLMs that read the full log. Over the same horizon, it matches knowledge-tracing models without a readable interface. In multi-session tutoring with simulated learners on ASSISTments 2017, planning with IMAGINE gains more on this memory than on a low-fidelity one in all ten protocol seeds. It lands above random sequencing on the former and below it on the latter. An LLM tutor reading the serialized READ interface also beats random sequencing, with both providers. Memory size is $O(1)$ in history length, versus $O(t)$ for logs.

1 INTRODUCTION

Memory is integral to LLM agents, and the systems built for it record history and answer the same question, namely what has happened so far. Paged context (Packer et al., 2023), consolidated summaries (Chhikara et al., 2025), decayed episodic stores (Zhong et al., 2024) and graph-indexed retrieval (Gutiérrez et al., 2024) differ in what they keep and how they find it again.

Tutoring is where a record falls short. The decision a tutor faces is which exercise to give next, and no record of past events answers it. Two quantities answer it. The current knowledge state of the learner is latent and must be inferred. The consequence of a candidate teaching action has not happened yet and must be predicted. A memory that can only report the past leaves both quantities to be reconstructed by the policy at every decision.

Three lines of work each supply part of the missing capability, yet each stops short. Agent memory reads and writes but does not look ahead. World models learn to roll an environment forward (Hafner et al., 2020), but that environment is not a person. Recent work extends the idea to user state (Luo et al., 2026) while keeping the state model and the memory separate. Learner modeling has predicted responses for three decades (Corbett & Anderson, 1995; Piech et al., 2015), but its models serve as external simulators rather than as the memory an agent carries. Probabilistic belief has been proposed as the content of agent memory (Liao et al., 2026), without a forward model.

This paper closes the gap in one state rather than by composition. The memory of a tutoring agent is a mental world model of the learner, a fixed-size belief state exposing three operations. WRITE is a

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

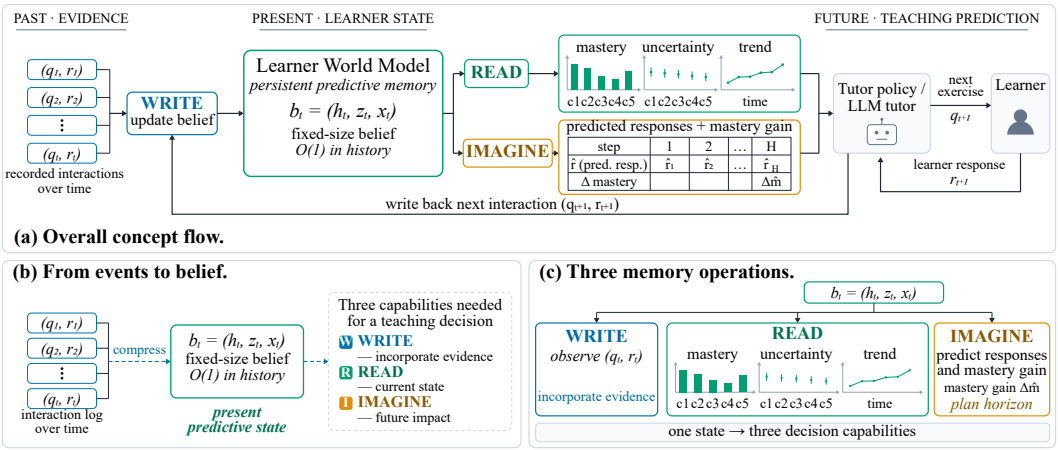


Figure 1: From recorded interactions to a world-model memory of the learner. (a) The closed loop of a teaching decision. The past is a growing log of interactions (q_i, r_i) in time order. WRITE folds each exercise and response into the present learner state, a fixed-size belief $b_t = (h_t, z_t, x_t)$ that is $O(1)$ in history length. READ exposes per-concept mastery with its uncertainty and trend. IMAGINE predicts the responses $\hat{r}_{1:H}$ and the mastery gain $\Delta\hat{m}$ of an H-step candidate sequence. The exercise q_{t+1} selected by the policy is logged with the response r_{t+1} from the learner as the next interaction. (b) The compression itself. An unbounded event log becomes one fixed-size predictive state, which serves the three capabilities needed for a teaching decision. (c) WRITE, READ and IMAGINE are three operations on one fixed-size state, not three stores

posterior update, READ returns per-concept mastery with uncertainty, and IMAGINE rolls a candidate teaching sequence forward inside the memory. The first two have counterparts in conventional agent memory; the third is what a log alone cannot support and what a teaching decision needs. Figure 1 shows the resulting loop. Evidence is written into one state. The state is read and imagined forward, and the exercise chosen from those two readouts yields the next piece of evidence.

The paper makes three contributions. **Memory as a learner world model** holds updating, reading and imagining in one fixed-size belief state. Three propositions show why such a belief keeps evidence that a fixed-budget log can lose and why planning on a poor memory can be worse than not planning. **A protocol for long-horizon tutoring** with simulated learners supplies the controls that make its comparisons interpretable. **Evidence** shows the memory 8.5–16.2 AUC points above retrieval and Bayesian Knowledge Tracing for the next 100 responses, at constant cost. READ supports personalization on all three corpora. On ASSISTments 2017, planning gains more on this memory than on a low-fidelity one in all ten seeds, and on the latter it falls below random sequencing.

2 Related Work

Agent memory. Current systems organize what to keep and how to retrieve it. Some systems page between context and an archival store (Packer et al., 2023), consolidate salient facts through explicit merge operations (Chhikara et al., 2025) or decay episodic entries on a forgetting curve (Zhong et al., 2024). Others index a graph for multi-hop retrieval (Gutiérrez et al., 2024) or link notes into a network rewritten on each insertion (Xu et al., 2025). Another approach generates rather than stores latent memory representations (Zhang et al., 2025). It addresses how memory is encoded, not whether memory can be simulated forward. These systems are measured on retrieval, test-time learning, long-range understanding and selective forgetting (Hu et al., 2025); none is measured on prediction. Memories whose content is probabilistic belief (Liao et al., 2026; Singh, 2026) have no rollout operation and are evaluated on dialogue and household tasks. None of these systems answers what happens under a candidate action.

World models over people. Latent-imagination agents plan by rolling a learned model of the environment forward (Hafner et al., 2020). A concurrent LLM agent sharpens such foresight with an

episodic and semantic memory of observed transitions (Zhang et al., 2026). Modeling a person this way is recent. A transition and observation model of user state supports lookahead over candidate utterances (Luo et al., 2026). A dialogue policy is optimized against an intrinsic model of the anticipated reaction of the user (He et al., 2025). These approaches address counseling and conversational search rather than education. They keep the state model separate from the memory of the agent. This paper removes that separation.

Learner modeling and instructional sequencing. Knowledge tracing predicts the next response from the interaction history, from two-state per-skill models (Corbett & Anderson, 1995) to recurrent (Piech et al., 2015) and attentive (Ghosh et al., 2020) variants. Sequencing has been posed as planning against such a model (Rafferty et al., 2016; Deng et al., 2025), and teaching policies are now learned rather than designed (Dinucu-Jianu et al., 2025). Reinforcement learning over simulated learners also has a long record of results that fail to transfer (Doroudi et al., 2019). Its evaluation practice is therefore treated carefully here. Prompted student personas make weak simulated learners. They score poorly on linguistic, behavioral and cognitive fidelity, and tuning closes only part of the gap (Scarlatos et al., 2026; Naous et al., 2026). In all of this work the learner model is an external simulator or predictor, not the memory the agent carries between sessions.

LLM tutors. Dialogue tutoring datasets, tuned tutoring models and benchmarks that score their pedagogy (Macina et al., 2023; LearnLM Team, 2024; Macina et al., 2025) evaluate single sessions or single turns. Agentic tutors have begun to carry a learner memory alongside static knowledge (Zhao et al., 2026), without asking it to predict. Persistent learner state is standard in symbolic intelligent tutoring systems but degenerates in LLM tutors into whatever fits the context window. A concurrent benchmark places tutoring agents in a 30-day relationship with knowledge-tracing-grounded simulated learners and finds that few model-and-harness combinations sustain teaching quality over that horizon (Lee et al., 2026). It measures agents rather than proposing a memory, and its finding motivates the mechanism studied here.

3 Memory as a learner mental world model

3.1 Setting and three operations

A learner interacts with exercises over time. At step t the tutor selects an exercise q_t from the history and its own randomness and observes a binary response r_t. The knowledge state s_t is latent and evolves as $s_{t+1} \sim P(\cdot \mid s_t, q_t)$, with $r_t \sim P(\cdot \mid s_t, q_t)$. This assumption is the standard one behind knowledge tracing, shared by two-state per-skill models and their neural successors (Corbett & Anderson, 1995; Piech et al., 2015).

An agent memory is a function $M_t = f(h_{1:t})$ of the interaction history under a capacity constraint. The constraint is either a token budget for a memory the tutor reads as text or a fixed dimension for a parameterized one. The memory exposes three operations:

$\textsc{write}(M, (q, r)) \to M'$	state update from one interaction
$\textsc{read}(M) \to (\hat{m}, \sigma)$	per-concept mastery and its uncertainty
$\textsc{imagine}(M, q_{1:H}) \to (\hat{r}_{1:H}, \Delta\hat{m})$	predicted trajectory of a candidate sequence

Every memory in this paper exposes these three calls, so a policy runs unchanged across the memories it is compared on. For one policy, conditions differ only in the memory and its readout. Log-based memories implement IMAGINE by assuming only that the learner does not change, so their predicted mastery change is zero by construction.

3.2 Why a belief state

Two properties separate a parameterized belief from a text log, both proved in Appendix B.

Proposition 1 (Sufficiency)**.** *Under the latent-state assumption and history-measurable scheduling (Appendix B), the belief $b_t = P(s_{t+1} \mid h_{1:t})$ satisfies $I(b_t; r_{t+1:t+H} \mid q_{t+1:t+H}) = I(h_{1:t}; r_{t+1:t+H} \mid q_{t+1:t+H})$ for every horizon H. A memory from which b_t is recoverable predicts future responses as well as the entire history does.*

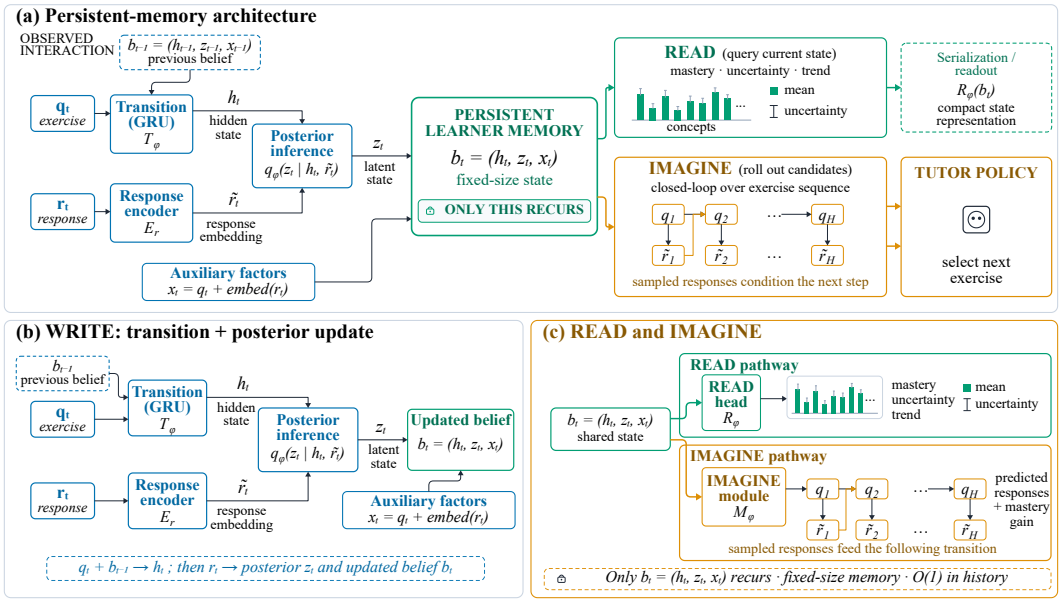


Figure 2: Implementation of the fixed-size world-model memory of the learner. (a) Architecture. The transition T_ϕ forms h_t from the previous belief state and the exercise q_t. The encoded response $\tilde{r}_t$ then determines the posterior latent z_t. Only $b_t = (h_t, z_t, x_t)$ serves as the previous state for the next WRITE; everything to its left is recomputed per interaction. That state feeds READ and closed-loop IMAGINE. The readout $R_\phi(b_t)$ serializes into the compact state a text-consuming tutor receives. In IMAGINE, the sampled responses condition the next rollout step. (b) The WRITE path. The exercise and previous belief state form h_t. The response then gives the posterior z_t and the updated belief. (c) The two query paths. The READ head sits on the shared state, and the IMAGINE module rolls a candidate sequence $q_{1:H}$ forward using its own sampled responses

Proposition 2 (Capacity dilemma of logs)**.** *Let a memory hold at most B tokens of the history, kept by truncation or by retrieval that ranks interactions by concept similarity to the query and breaks ties toward recent ones. For some learner processes the full log requires $\Omega(t)$ tokens. On these processes, for any fixed B, the retained selection can miss the one interaction that determines the next response and then carries no information about it. A belief of dimension independent of t remains sufficient on the same processes.*

The construction behind Proposition 2 is a learner whose early interactions determine which of two latent branches holds, and whose later interactions are identical across branches. Neither selection retains that evidence, whereas a belief carries the branch posterior forward at constant size.

Prediction error in the memory also bounds planning performance. Let $\hat{b}$ be the belief implied by a memory, ε be $\|\hat{b} - b\|_1$ and δ bound the total-variation error of its one-step response prediction. For bounded gains, the standard simulation argument shows that a plan greedy against the imagined value loses at most $C(H)(\varepsilon + H\delta)$ in learning gain, with C linear in plan length (Kearns & Singh, 2002; Kaelbling et al., 1998). The reverse reading matters in practice. When the forward model of a memory is poor, planning can be worse than not planning. Section 5 reports that outcome.

Proposition 3 (Planning under an imperfect memory)**.** *For any $\eta > 0$ there is a learner process and a memory with response-model error $\delta < \eta$ for which the plan maximizing the imagined gain achieves less expected true gain than random sequencing.*

3.3 INSTANTIATION

The memory is instantiated as a recurrent state-space model of the learner (Figure 2). Only three tensors carry the state from one WRITE to the next. They are a deterministic recurrent state h_t, the embedding x_t of the interaction just observed and a stochastic latent z_t that feeds the mastery readout. WRITE runs the transition T_ϕ, a recurrent cell mapping the previous belief and the embedded

exercise q_t to h_t. The response is encoded by E_r and enters the posterior $q_\phi(z_t \mid h_t, \tilde{r}_t)$. The exercise thus moves the state before the response is seen, and the response then sharpens it. Two choices are specific to the education setting. The previous exercise and response enter the cell in one step, bound into x_{t-1}. Splitting them leaves the cell to re-associate a response with the exercise from the preceding step. Reconstruction uses a separate head. With one-bit observations, a shared decoder can copy the response that leaks through the posterior latent and degrade the prior path. Removing either choice costs more than any capacity change tested (Appendix E).

READ applies R_ϕ to b_t for per-concept mastery and adds uncertainty and trend from bounded stored statistics. The serialized output is what a text-consuming tutor receives in place of a log. IMAGINE rolls the same state forward under a candidate sequence with the loop closed. Each step samples a response from the predicted distribution, binds the response with its exercise into the next x and feeds that back through the transition. The rollout thus compounds its own predictions as in deployment. Training matches this regime by adding closed-loop overshooting to the one-step objective. Each operation costs $O(1)$ in the number of interactions written so far, whereas a memory that keeps the log costs $O(t)$.

4 Evaluation protocol

Data and learners. Interaction streams come from three public knowledge-tracing corpora, ASSISTments 2009, ASSISTments 2017 and the diagnostic questions of the NeurIPS 2020 Education Challenge (NIPS34; Wang et al., 2020). Learners are split disjointly into train, validation and test sets. Tests and per-learner statistics below aggregate the windows of a learner first.

Simulated learners. An attentive knowledge-tracing model (AKT) trained on real interactions simulates each learner and decides correctness. The tutor sees the same binary responses as in a real system. Using a fitted model rather than an LLM playing a student avoids the failure modes reported for prompted student personas. This choice follows the knowledge-tracing-grounded learners of Lee et al. (2026) rather than the LLM-driven learners of Gao et al. (2025). This setup gives a learner model fitted to real response data, but it omits everything beyond correctness (Section 6).

Protocol. Each learner is warmed up on $t_0 = 50$ observed interactions written into the memory, then taught for five sessions of 10 exercises. A probe set measures mastery before and after teaching. Learning gain is the change in predicted probe correctness.

Four controls. Probe items are removed from the pool that the tutor may schedule, because otherwise a policy can teach the test. A difficulty-matched control draws random exercises near the mean difficulty of the greedy policy, so that a difficulty effect is not mistaken for personalization. Probe scoring uses models other than the memory being tested, and results under two separately trained judges are reported separately rather than averaged. The primary measurement is the probe change of the simulated learner, the closest available measure of what is learned. The external judges test whether a result survives a change of measuring instrument.

Conditions. The fidelity comparison covers retrieval, a Bayesian Knowledge Tracing state (BKT-in-prompt, named for how an LLM tutor reads it), the proposed world-model memory and two knowledge-tracing models. The cost comparison adds full context. The tutor axis includes random sequencing, a difficulty-matched control, a greedy policy on the weakest concept, a random-shooting planner over IMAGINE, a focus-m policy that rotates over the m weakest concepts and an LLM tutor that plans one session per call from the serialized memory. In the main comparison, every tutor runs on the world-model memory, and the planner also runs on BKT-in-prompt.

5 Results

5.1 Memory fidelity (E1)

Table 1 reports how well each memory predicts the future responses of a learner from an observed prefix and the sequence of exercises that follows. Rollout AUC and calibration error at horizon h

Table 1: Memory fidelity at rollout horizon 100. Bold marks the best value per corpus and column, not the proposed method. AKT and DKT are knowledge-tracing models run through the same evaluator. They predict close to the world-model memory but use no readable interface in this table. The other three rows are memories that a tutor can read. Retrieval loses item-level discrimination and calibration, whereas BKT-in-prompt loses per-learner discrimination. Each ± is one standard deviation over seeds 42–44

Dataset	Memory	$AUC_{h=100}$	$ECE_{h=100}$	$r_{h=100}$
ASSIST2009	Mental world model (ours)	**0.724** ±0.006	0.040 ±0.006	0.824 ±0.018
	AKT	0.722 ±0.002	0.042 ±0.020	0.761 ±0.022
	DKT	0.720 ±0.004	0.035 ±0.014	**0.828** ±0.019
	BKT-in-prompt	0.639	**0.022**	0.247
	Retrieval	0.629	0.092	0.780
ASSIST2017	Mental world model (ours)	**0.741** ±0.002	0.018 ±0.001	**0.850** ±0.010
	AKT	0.715 ±0.003	0.039 ±0.009	0.808 ±0.015
	DKT	0.724 ±0.002	**0.018** ±0.002	0.796 ±0.011
	BKT-in-prompt	0.579	0.055	0.144
	Retrieval	0.590	0.096	0.723
NIPS34	Mental world model (ours)	0.774 ±0.000	0.022 ±0.003	**0.904** ±0.001
	AKT	0.772 ±0.001	**0.011** ±0.004	0.889 ±0.002
	DKT	**0.774** ±0.001	0.020 ±0.007	0.897 ±0.004
	BKT-in-prompt	0.632	0.143	0.755
	Retrieval	0.684	0.067	0.843

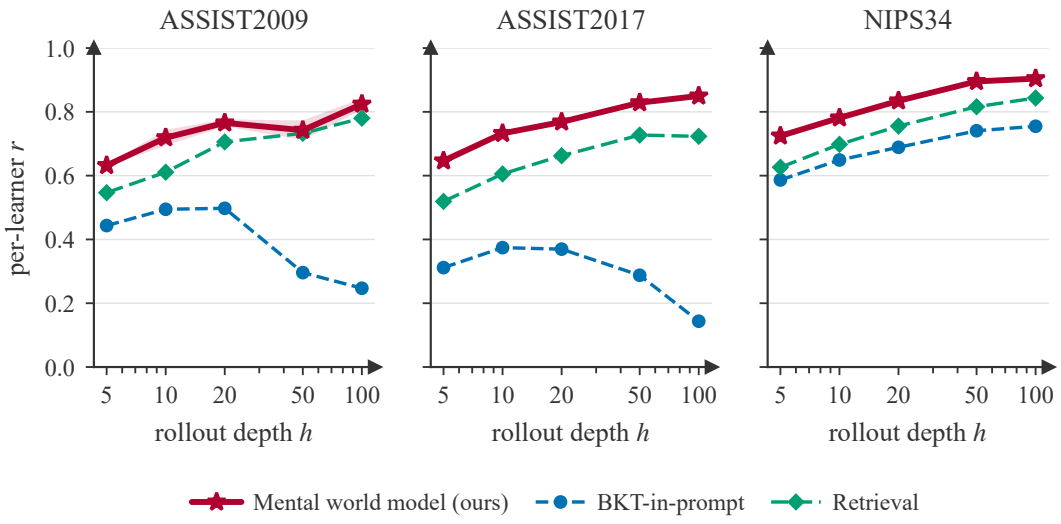


Figure 3: Per-learner discrimination against rollout depth. The band around the world-model memory is one standard deviation over three seeds

pool all predictions of the first h steps of a closed-loop rollout after 50 observed interactions. Per-learner correlation r is the correlation across learners between predicted and observed accuracy over those h steps.

Against the readable memories, the margins are wide. At horizon 100, the world-model memory leads BKT-in-prompt by 8.5, 16.2 and 14.2 points and retrieval by 9.6, 15.2 and 9.0 points in rollout AUC. The two memories fail differently. On the ASSISTments corpora, BKT-in-prompt matches retrieval in item-level but not in per-learner discrimination. Its per-learner correlation is 0.247, 0.144 and 0.755, against 0.824, 0.850 and 0.904 for the world-model memory. Retrieval keeps per-learner correlation (0.780, 0.723, 0.843) but loses item-level discrimination and calibration. Its expected calibration error reaches 0.092 and 0.096 on the ASSISTments corpora. Of the three readable memories, only the world-model memory keeps both kinds of discrimination.

The world-model memory also matches or leads the knowledge-tracing models, which predict well but expose no native interface a tutor can read. On ASSISTments 2017, it leads AKT and DKT by 2.7 and 1.7 points of rollout AUC, and on ASSISTments 2009 and NIPS34 they come within one point of it (0.722 and 0.720, 0.772 and 0.774). A state built to be read and rolled forward thus predicts as well as models built only to predict, and it also answers the two questions a teaching decision asks.

In Figure 3, per-learner discrimination of the world-model memory rises overall with rollout depth on all three corpora, from 0.63 to 0.82, 0.65 to 0.85 and 0.72 to 0.90. On the ASSISTments corpora, the curve for BKT-in-prompt peaks by $h = 20$ and then falls, ending at 0.25 and 0.14. There, the world-model memory keeps this discrimination over a long rollout and BKT-in-prompt loses it.

Rollout fidelity depends on the training objective more than on capacity. Tuning the training on ASSISTments 2017 raises validation rollout AUC by $+0.025$, whereas increasing latent width alone yields $+0.0005$. Removing the overshoot term from the first row of Table 13 costs 0.047 ($p = 0.002$), a loss in each of three seeds (Appendix D). The tuning lowers teacher-forced AUC, so its gain is specific to closed-loop rollout. The following results are obtained without that tuning.

5.2 Tutoring value (E2)

Table 2 and Figure 4 compare teaching policies against random sequencing over the same learners. Against the difficulty-matched control, greedy on the weakest concept yields $+0.043$ in learning gain on ASSISTments 2017 ($p < 10^{-12}$, $n = 162$), $+0.017$ on ASSISTments 2009 ($p = 0.027$, $n = 167$) and $+0.088$ on NIPS34 ($p < 10^{-35}$, $n = 273$). A policy that knows which concepts are weak beats random choice at matched mean difficulty on all three corpora. Against random sequencing, it is indistinguishable on ASSISTments 2017 (Table 2) and yields $+0.019$ on ASSISTments 2009 and $+0.096$ on NIPS34 (Appendix C).

The value of imagination depends on the memory it runs on. On ASSISTments 2017, the same planner gains $+0.025$ more on the world-model memory than on BKT-in-prompt ($p < 10^{-8}$). The difference is positive in each of ten protocol seeds and significant with either learners or seeds as the unit (Appendix C). Against random sequencing, the planner scores $+0.013$ on the world-model memory ($p = 0.002$, and $p = 0.037$ with ten seeds as the unit) and -0.011 on BKT-in-prompt ($p = 0.016$), worse than not planning. Proposition 3 shows it can happen.

On ASSISTments 2009 and NIPS34, the planner on the world-model memory stays ahead only within noise ($+0.003$ with $p = 0.63$ and $+0.004$ with $p = 0.45$). The two memories differ most in fidelity on ASSISTments 2017, by 16.2 points of rollout AUC and 0.71 in per-learner correlation, against at most 14.2 points and 0.58 on the other corpora. Across the three corpora, the planners separate only where fidelity differs most, and there planning on the poor memory falls below not planning. DKT with a mastery readout is a second faithful memory, and planning on it also beats random sequencing (Appendix C).

5.3 An LLM tutor on the interface (E3)

An LLM planning one session per call from the serialized memory beats random sequencing by $+0.037$ ($p = 0.014$) with one provider and $+0.063$ ($p = 0.003$) with a second. Both margins exceed that of every scripted policy in Table 2. The first estimate uses one batch of the tutoring protocol of Table 2 and pairs its 22 learners against the random arm over three seeds. Running the same prompt over eight batches at one seed covers 162 paired learners and yields an estimate of $+0.036$ with $p = 9 \times 10^{-5}$. Successive blocks of 64 windows do not differ ($F = 0.96$, $p = 0.46$). Clustering the standard error by block still gives $p = 0.006$.

No LLM condition falls back to greedy (Appendix F).

Supplying imagined outcomes to the LLM tutor does not help. Rolling out heuristic candidate sessions and reporting their predicted gains changes the read-only gain by -0.013 and -0.025 for the two providers. Rolling out the plan drafted by the LLM itself and inviting revision changes it by -0.004 and -0.038. No increment is significant. All four have the same sign across two providers and two ways of presenting imagined outcomes.

Table 2: Multi-session tutoring on ASSISTments 2017, measured by the simulated learner. Each row is the learner-level paired difference against random sequencing over the same learners; $^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$

Policy	Δ gain vs. random	SEM	n
Difficulty-matched control	-0.042***	0.006	162
Greedy on weakest concept (read only)	+0.001	0.006	162
Planner on BKT-in-prompt (read + imagine)	-0.011*	0.005	162
Planner on world-model memory (read + imagine)	+0.013**	0.004	162
LLM tutor on world-model memory	+0.037*	0.014	22
LLM tutor, second provider	+0.063**	0.019	22

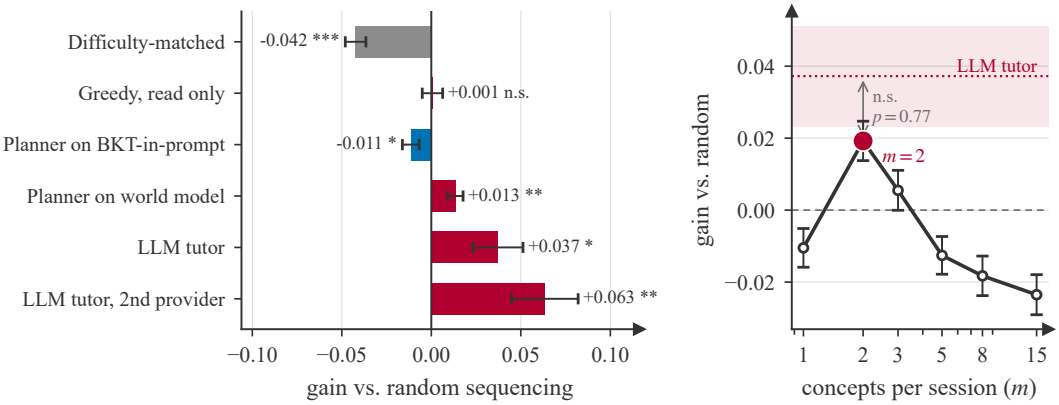


Figure 4: The teaching value of a memory, measured against random sequencing over the same learners. In the left panel, color marks the memory a policy reads (gray for none), and the same planner yields $+0.013$ on the world-model memory and -0.011 on BKT-in-prompt. The right panel plots gain against the number of concepts practiced in a session. Gain peaks at $m = 2$, where the scripted policy is within noise of the LLM tutor. Bars and points show learner-level means, and whiskers show one standard error

What the LLM does with the memory is simple to describe. Figure 4 (right) shows a sweep over a scripted policy that rotates over the m weakest concepts. Gain against random sequencing is single-peaked. It is -0.010 at $m = 1$, $+0.019$ at $m = 2$ ($p < 10^{-3}$), $+0.006$ at $m = 3$, then -0.013 at $m = 5$ and -0.024 at $m = 15$. Concentrating on one concept and spreading over 15 both fall below random sequencing. The LLM tutor is not distinguishable from the scripted policy at the peak ($+0.004$, $p = 0.77$). The LLM tutor produces sequences with 0.59 distinct exercises per slot, a diversity between the values for $m = 2$ and $m = 3$. Given a memory that identifies weak concepts, the focus policy comes within noise of the LLM tutor at the right focus level. Appendix F runs the same prompt on BKT-in-prompt.

5.4 Memory cost (E4)

Table 3 and Figure 5 report the cost of one read as a learner accumulates history. The world-model memory serializes to 94 tokens at five concepts regardless of history length. Full context grows from 550 tokens at 50 interactions to 2200 at 200. Retrieval reads a digest of at most 64 tokens and stores the whole log to build the digest. Its read cost is therefore bounded, but its storage cost is not.

Cost and fidelity distinguish the memories. Full context and retrieval share one log and one predictor, but full context requires an order of magnitude more prompt tokens. The world-model memory reaches 0.741 rollout AUC at 94 tokens against 0.590 at 2200 tokens for full context. Read by either of two LLMs, the full context of 50 interactions (550 to 851 tokens) reaches AUC 0.580 to 0.643, against 0.699 to 0.786 for the seed-42 world-model memory (Appendix C).

Table 3: Memory cost against history length on ASSISTments 2017. Retrieval reads a bounded digest but keeps the whole log to build it. Tokens are word and symbol counts. State counts the belief state or the log (Appendix A)

	Prompt tokens per read			State	
Memory	@50	@200	growth	@200 (B)	growth
Mental world model (ours)	94	94	$O(1)$	1184	$O(1)$
BKT-in-prompt	64	64	$O(1)$	412	$O(1)$
Retrieval	53	64	$O(1)$	2400	$O(t)$
Full context	550	2200	$O(t)$	2400	$O(t)$

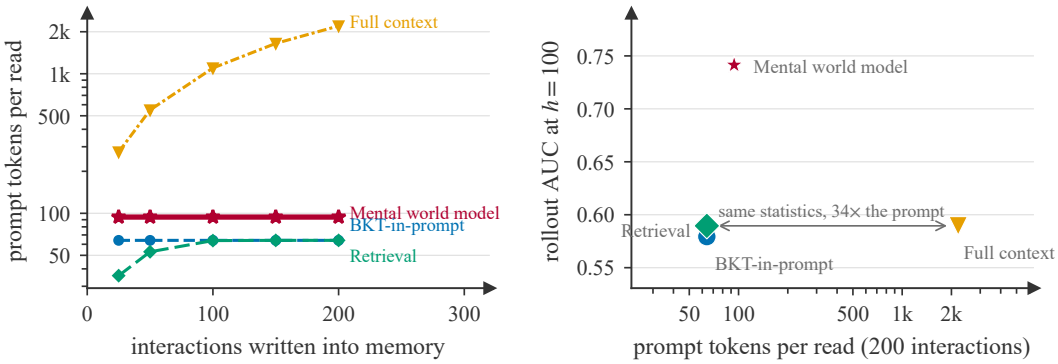


Figure 5: Prompt cost of one read against interactions written (left) and the cost-fidelity plane (right), with tokens at 200 interactions and rollout AUC at $h = 100$. Full context carries the same log statistics as retrieval at an order of magnitude more prompt tokens

6 Limitations

The learners are simulated, and simulation bounds what the results can show. Responses are binary correctness from a fitted knowledge-tracing model, so explanation quality, motivation and affect go unmeasured. Under the probe of a DKT learner in place of AKT, the planner still gains 0.011 more on the world-model memory than on BKT-in-prompt at the learner level ($p = 0.024$, Table 9). The domain is mathematics exercise sequencing on three corpora, with the concept vocabulary their tagging supplies.

Forgetting is not modeled. Every model receives interactions in order but not the time between them, and the five sessions of the protocol are consecutive blocks of 10 exercises.

7 Conclusion

A tutoring agent needs a memory that answers two questions, what a learner knows and what a teaching action does. Both concern learner states, not interaction records. As a mental world model of the learner, the memory answers both questions from one fixed-size belief state. It predicts the next 100 responses 8.5–16.2 AUC points better than retrieval and BKT-in-prompt, at constant cost, and better than two LLMs that read the full log. READ supports personalization on all three corpora, whereas IMAGINE supports planning on the corpus where the two memories differ most in fidelity. There, the same planner gains more on this memory than on a low-fidelity one in all ten seeds. On that corpus and with the AKT learner, planning on this memory beats random sequencing under every judge.

AI use statement

The authors use generative AI tools in several parts of this work. They assist in writing the proofs of Appendix B and in narrowing the statement of Proposition 2. They help design experiments and give feedback on methodology, including the planner baseline on a knowledge-tracing memory and

the comparison of units of analysis in Appendix C. They implement parts of the methods and of the code that runs the experiments and analyses, help interpret the results, and draft and edit the manuscript. Reviews of the manuscript by a language model inform revisions to its claims. All AI-assisted work is reviewed by the authors. The main results are recomputed from the recorded outputs by a verification script, and the incremental memory backend is tested against the batched model it wraps. The authors take responsibility for the content of this work, including its text, claims and artifacts.

Ethics statement

The experiments use three public knowledge-tracing corpora released for research use that contain no personally identifying information. No new data are collected from human subjects, and no human subject interacts with any system described here. The tutoring policies teach learners simulated by models fitted to those corpora. Simulation makes long-horizon comparison affordable and bounds every empirical claim. Nothing reported here establishes that a policy helps a real learner (Section 6). Two concerns arise from deploying a memory of this kind with real learners and are not resolved by this study. A persistent inferred model of what a learner knows is a sensitive record. It is held longer and infers more than the log it replaces. A policy optimized against a fitted learner model can exploit that model instead of teaching the learner, a risk tested here through separate judges. Appendix C reports every policy of Table 2 under a same-architecture judge and a cross-architecture judge.

Reproducibility statement

The main results of this paper are recomputed from the recorded output files by a single script. The script will be released together with the code, the configurations, the protocol and the seeds. The corpora are public. Appendix A states the objective, the hyperparameters and which trained model fills each role. Appendix C gives the seeds, the paired-test convention and the judge-by-policy report that Section 4 commits to. Appendix F names the two tutoring models and gives their prompts verbatim together with the fallback counts. The propositions are proved in Appendix B.

References

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready AI agents with scalable long-term memory. *arXiv preprint arXiv:2504.19413*, 2025.

Albert T. Corbett and John R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction*, 4(4):253–278, 1995.

Yang Deng, Zifeng Ren, An Zhang, and Tat-Seng Chua. Towards goal-oriented intelligent tutoring systems in online education. *ACM Transactions on Information Systems*, 2025. arXiv:2312.10053.

David Dinucu-Jianu, Jakub Macina, Nico Daheim, Ido Hakimi, Iryna Gurevych, and Mrinmaya Sachan. From problem-solving to teaching problem-solving: Aligning LLMs with pedagogy using reinforcement learning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, 2025. arXiv:2505.15607.

Shayan Doroudi, Vincent Aleven, and Emma Brunskill. Where's the reward? a review of reinforcement learning for instructional sequencing. *International Journal of Artificial Intelligence in Education*, 29:568–620, 2019.

Weibo Gao, Qi Liu, Linan Yue, Fangzhou Yao, Rui Lv, Zheng Zhang, Hao Wang, and Zhenya Huang. Agent4Edu: Generating learner response data by generative agents for intelligent education systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025. arXiv:2501.10332.

Aritra Ghosh, Neil Heffernan, and Andrew S. Lan. Context-aware attentive knowledge tracing. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2020.

Bernal Jiménez Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. HippoRAG: Neurobiologically inspired long-term memory for large language models. In *Advances in Neural Information Processing Systems*, 2024.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2020.

Tao He, Lizi Liao, Ming Liu, and Bing Qin. Simulating before planning: Constructing intrinsic user world model for user-tailored dialogue policy planning. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2025. arXiv:2504.13643.

Yuanzhe Hu, Yu Wang, and Julian McAuley. Evaluating memory in LLM agents via incremental multi-turn interactions. *arXiv preprint arXiv:2507.05257*, 2025.

Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1–2):99–134, 1998.

Michael Kearns and Satinder Singh. Near-optimal reinforcement learning in polynomial time. *Machine Learning*, 49:209–232, 2002.

LearnLM Team. LearnLM: Improving gemini for learning. *arXiv preprint arXiv:2412.16429*, 2024.

Unggi Lee, Sookbun Lee, Yeil Jeong, Eunjoo Lee, Minchul Shin, and Hoilym Kwon. EduClaw-Bench: A long-horizon benchmark for pedagogical LLM agents with simulated learners. *arXiv preprint arXiv:2608.03206*, 2026.

Junfeng Liao, Qizhou Wang, Jianing Zhu, Bo Du, Rui Yan, and Xiuying Chen. Belief memory: Agent memory under partial observability. *arXiv preprint arXiv:2605.05583*, 2026.

Jiani Luo, Xiaoyan Zhao, Yang Zhang, Shuyi Miao, Bingbing Xu, Stefan Konigorski, and Tat-Seng Chua. Know you before you speak: User-state modeling for LLM personalization in multi-turn conversation. *arXiv preprint arXiv:2605.24647*, 2026.

Jakub Macina, Nico Daheim, Sankalan Pal Chowdhury, Tanmay Sinha, Manu Kapur, Iryna Gurevych, and Mrinmaya Sachan. MathDial: A dialogue tutoring dataset with rich pedagogical properties grounded in math reasoning problems. In *Findings of the Association for Computational Linguistics: EMNLP*, 2023.

Jakub Macina, Nico Daheim, Ido Hakimi, Manu Kapur, Iryna Gurevych, and Mrinmaya Sachan. MathTutorBench: A benchmark for measuring open-ended pedagogical capabilities of LLM tutors. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, 2025. arXiv:2502.18940.

Tarek Naous, Philippe Laban, Wei Xu, and Jennifer Neville. Flipping the dialogue: Training and evaluating user language models. In *International Conference on Learning Representations*, 2026. arXiv:2510.06552.

Charles Packer, Vivian Fang, Shishir G. Patil, Kevin Lin, Sarah Wooders, and Joseph E. Gonzalez. MemGPT: Towards LLMs as operating systems. *arXiv preprint arXiv:2310.08560*, 2023.

Chris Piech, Jonathan Bassen, Jonathan Huang, Surya Ganguli, Mehran Sahami, Leonidas Guibas, and Jascha Sohl-Dickstein. Deep knowledge tracing. In *Advances in Neural Information Processing Systems*, 2015.

Anna N. Rafferty, Emma Brunskill, Thomas L. Griffiths, and Patrick Shafto. Faster teaching via POMDP planning. *Cognitive Science*, 40(6):1290–1332, 2016.

Alexander Scarlatos, Jaewook Lee, Simon Woodhead, and Andrew Lan. Simulated students in tutoring dialogues: Substance or illusion? In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics*, 2026. arXiv:2601.04025.

Pranav Singh. Nous: A predictive world model for long-term agent memory. *arXiv preprint arXiv:2606.22030*, 2026.

Zichao Wang, Angus Lamb, Evgeny Saveliev, Pashmina Cameron, Yordan Zaykov, José Miguel Hernández-Lobato, Richard E. Turner, Richard G. Baraniuk, Craig Barton, Simon Peyton Jones, Simon Woodhead, and Cheng Zhang. Instructions and guide for diagnostic questions: The NeurIPS 2020 education challenge. In *arXiv preprint arXiv:2007.12061*, 2020.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-MEM: Agentic memory for LLM agents. In *Advances in Neural Information Processing Systems*, 2025. arXiv:2502.12110.

Guibin Zhang, Muxin Fu, and Shuicheng Yan. MemGen: Weaving generative latent memory for self-evolving agents. *arXiv preprint arXiv:2509.24704*, 2025.

Xuan Zhang, Wenxuan Zhang, See-Kiong Ng, and Yang Deng. Self-evolving world models for LLM agent planning. *arXiv preprint arXiv:2606.30639*, 2026.

Bingxi Zhao, Jiahao Zhang, Xubin Ren, Zirui Guo, Tianzhe Chu, Yi Ma, and Chao Huang. DeepTutor: Towards agentic personalized tutoring. *arXiv preprint arXiv:2604.26962*, 2026.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. MemoryBank: Enhancing large language models with long-term memory. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2024.

A Model and training details

This appendix specifies the memory and its training objective and identifies every trained model used in an experimental role.

State and operations. The state is $b_t = (h_t, z_t, x_t)$, a deterministic recurrent vector h_t of width 200, a stochastic latent z_t of width 32 and the embedding x_t of the interaction just observed. Exercises and responses embed into width 64 and bind into one interaction vector, $x_t = q_t + E_r(r_t)$. WRITE runs a gated recurrent cell T_ϕ on (z_{t-1}, q_t, x_{t-1}) to give h_t, forms a diagonal Gaussian prior $p_\phi(z_t \mid h_t)$ and posterior $q_\phi(z_t \mid h_t, \tilde{r}_t)$, and samples z_t from the posterior in training but keeps the posterior mean in evaluation and tutoring. A prediction head over (h_t, z_t, q_t) returns the response logit, where z_t comes from the prior. A separate head reconstructs the observed response from the posterior path, and a third head over (h_t, z_t) returns per-concept mastery, which READ serializes. The state column of Table 3 counts this belief state. Serving the uncertainty and the trend additionally retains the posterior spread and the mastery vectors of the last 21 steps, both bounded and thus also $O(1)$ in history length. For the log-based memories, the column counts each interaction as three 4-byte integers for its exercise, concept and response. The uncertainty is the standard deviation of mastery over eight samples of z_t from the posterior, and the trend is the change in mastery over the last 20 steps. IMAGINE rolls the same cell forward on responses it samples itself.

Objective. Training minimizes, as a mean over the unmasked steps of a batch,

$$\mathcal{L} = \mathrm{BCE}(\hat{r}, r) + \lambda_{\mathrm{rec}}\,\mathrm{BCE}(\hat{r}^{\mathrm{post}}, r) + \beta_{\mathrm{KL}}\,\overline{\mathrm{KL}} + \lambda_{\mathrm{os}}\,\mathcal{L}_{\mathrm{os}} + \lambda_m\,\mathrm{BCE}(\hat{m}, r),$$

where $\hat{r}$ is the prediction head, $\hat{r}^{\mathrm{post}}$ the reconstruction head and $\hat{m}$ the mean of the mastery logits for the concepts carried by the exercise. The divergence term is balanced,

$$\overline{\mathrm{KL}} = \alpha\,\mathrm{KL}(\mathrm{sg}[q_\phi] \,\|\, p_\phi) + (1-\alpha)\,\mathrm{KL}(q_\phi \,\|\, \mathrm{sg}[p_\phi])\,,$$

with sg denoting stop-gradient, so the prior is pulled toward the posterior four times as strongly as the reverse. The term is clamped from below at the free-bits floor.

Closed-loop overshooting. $\mathcal{L}_{\mathrm{os}}$ trains the state to stay predictive when rolled forward. Four start points are drawn uniformly per batch. From each, the cell runs $d = 1, \ldots, D$ steps without observing a response. At each step, the prior supplies z, and the prediction head supplies a logit scored against the recorded response. The divergence between the posterior of the recorded sequence at that step and the rolled-out prior is added under the same free-bits floor. The next exercise binds with a response sampled from the predicted distribution, not with the recorded response. The resulting interaction vector conditions the following step. This term differs from the one-step loss by scoring against recorded responses while conditioning on sampled ones.

Hyperparameters. The reported configuration uses $\lambda_{\text{rec}} = 0.3$, $\beta_{\text{KL}} = 0.1$, $\alpha = 0.8$, a free-bits floor of 1.0 nat, $D = 5$, $\lambda_{\text{os}} = 0.3$, four overshoot start points, $\lambda_m = 0.1$ and dropout 0.2. Optimization uses Adam with a learning rate of 10^{-3}, a batch size of 64 sequences and gradient-norm clipping at 5.0. Training runs at most 100 epochs and stops early with patience 10 on single-step validation AUC. Rollout evaluation averages each predicted probability over 10 sampled trajectories and computes calibration error over 10 equal-width bins. Appendix D sweeps $|z|$, the hidden width, β_{KL}, D, λ_{os}, λ_{rec}, dropout and the learning rate in two passes, the first around this setting. Appendix E turns off the interaction binding and the separate reconstruction head. Every neural knowledge-tracing baseline uses the same embedding width and dropout and trains under the same optimizer, schedule and stopping rule. DKT is a single-layer LSTM with the hidden width of the world-model memory. AKT stacks four attention blocks of four heads each, with feed-forward width 256, and adds an L2 penalty of weight 10^{-5} on its item-difficulty scalars. BKT-in-prompt fits its four parameters per concept with 20 iterations of expectation maximization on the training split. The retrieval digest reports the eight concepts on which the learner is least accurate. The architectures still differ in size. On ASSISTments 2017 the world-model memory has 688,481 parameters, against 526,353 for DKT and 433,772 for AKT. Increasing the latent width of the world-model memory alone yields $+0.0005$ in rollout AUC (Appendix D).

Data. Each corpus is reduced to one table containing learner, exercise, concept set and response, with rows ordered by time. Sequences are cut into windows of at most 200 interactions, and each window starts from an empty memory. An exercise carries at most four concepts on the ASSISTments corpora and at most eight on NIPS34. Learners are split disjointly into 70%, 10% and 20% for training, validation and test under split seed 42.

Roles of the trained models. In the tutoring runs, the memory under test is the world model trained with seed 42. The simulated learner that decides correctness is the AKT model trained with seed 42, so no policy is scored by the model it plans against. The same-architecture judge is a world model trained with seed 43, and the cross-architecture judge is an AKT model trained with seed 43. Each protocol seed repeats the tutoring loop, not the training of any of these models.

Planner. The random-shooting planner uses IMAGINE. It samples 32 candidate sessions of 10 exercises from the pool at each session start, rolls each forward four times on the memory and schedules the candidate with the largest mean predicted mastery gain. A session is 10 exercises, a run is five sessions and the probe is 50 held-out items. The loop teaches windows in batches of up to 64, at most eight batches per run.

B PROOFS

Throughout, $h_{1:t} = (q_1, r_1, \ldots, q_t, r_t)$ is the interaction history, s_{t+1} is the latent knowledge state after the t-th interaction and $b_t = P(s_{t+1} \mid h_{1:t})$ is the belief. The future exercises and responses are $Q = q_{t+1:t+H}$ and $R = r_{t+1:t+H}$. Two assumptions from Section 3 are used throughout. *Latent-state dynamics.* $s_{t+1} \sim P(\cdot \mid s_t, q_t)$ and $r_t \sim P(\cdot \mid s_t, q_t)$, so given s_{t+1} and Q the future responses R are conditionally independent of the past. *History-measurable scheduling.* Q is a function of $h_{1:t}$ and exogenous randomness only, so scheduling adds no information about s_{t+1} beyond what $h_{1:t}$ already carries. In the tutoring experiments the second assumption holds by construction for every exercise sequence fixed before any of its responses is observed, since every tutor chooses from an exercise pool fixed before teaching and sees the learner only through the memory (Section 4). Such sequences include each candidate that IMAGINE rolls forward and each session that the planner or the LLM tutor plans.

B.1 PROPOSITION 1

Proof. One direction is immediate. Because b_t is a deterministic function of $h_{1:t}$, the data-processing inequality for conditional mutual information gives $I(b_t; R \mid Q) \leq I(h_{1:t}; R \mid Q)$.

For the converse, condition on Q and marginalize over the latent state,

$$P(R = \rho \mid h_{1:t}, Q) = \sum_s P(s_{t+1} = s \mid h_{1:t}, Q)\, P(R = \rho \mid s_{t+1} = s, Q)$$
$$= \sum_s b_t(s)\, P(R = \rho \mid s_{t+1} = s, Q),$$

where the first equality drops $h_{1:t}$ from the second factor by the latent-state assumption, and the second replaces $P(s_{t+1} \mid h_{1:t}, Q)$ with $P(s_{t+1} \mid h_{1:t}) = b_t$ by history-measurable scheduling. The right-hand side depends on $h_{1:t}$ only through the vector b_t, so $h_{1:t} \to b_t \to R$ is a Markov chain given Q, and therefore $I(h_{1:t}; R \mid Q, b_t) = 0$. The chain rule gives

$$I(h_{1:t}; R \mid Q) = I(b_t; R \mid Q) + I(h_{1:t}; R \mid Q, b_t) = I(b_t; R \mid Q).$$

If a memory $M = f(h_{1:t})$ admits $b_t = g(M)$, then $I(h_{1:t}; R \mid Q) \geq I(M; R \mid Q) \geq I(b_t; R \mid Q)$ by two applications of data processing. The endpoints are equal, so M predicts future responses as well as the whole history does. □

Section 5 scores learned states on the prediction this result guarantees.

B.2 Proposition 2

A log-based memory meets a token budget in two ways, and the statement concerns both. *Truncation* retains the most recent interactions, and *query-driven retrieval* retains the interactions whose concepts are most similar to the concept of a query formed from the exercise about to be scheduled.

Proof. Let the latent type be $\theta \in \{A, B\}$, drawn uniformly, and use three exercise families over concepts c_0, c_1, c_2. A single *diagnostic* exercise q^* on c_0 is answered correctly if and only if $\theta = A$. A *filler* family on c_1 is answered correctly with probability $1/2$, independently of θ and of every other response. A *probe* exercise $q^\dagger$ on c_2 is answered correctly if and only if $\theta = A$. The learner attempts q^* once, at position j, and filler exercises at all remaining positions in $1, \ldots, t$.

The full log costs $\Omega(t)$. The $t - 1$ filler responses are i.i.d. fair coins, so $H(h_{1:t}) \geq t - 1$ bits. Any lossless encoding of the history therefore needs $\Omega(t)$ bits and hence $\Omega(t)$ tokens at any fixed rate.

Any fixed budget loses the branch. A budget of B tokens holds at most $m = m(B)$ interactions, where m is independent of t. Take $t > m + 1$. Truncation retains positions $t - m + 1, \ldots, t$, so setting $j = 1$ drops q^*. Query-driven retrieval forms its query from the exercise $q^\dagger$ on c_2 and ranks the c_1 fillers and the single c_0 diagnostic by similarity to c_2. For $t > m + 1$, at least m fillers are available. For any concept similarity, the construction takes c_1 to be a concept most similar to c_2 other than c_2 itself, so the similarity never ranks c_0 above c_1. With ties broken toward recent interactions, the retained selection again excludes q^*. In both cases, the retained selection S is a function of filler responses alone, which are independent of θ, so $I(S; r_{t+1} \mid q_{t+1} = q^\dagger) = 0$.

A constant-size belief keeps the branch. Take $b_t = P(\theta = A \mid h_{1:t})$, one scalar for every t. The response at position j determines θ, so $b_t \in \{0, 1\}$ and $I(b_t; r_{t+1} \mid q_{t+1} = q^\dagger) = H(\theta) = 1$ bit, which equals $I(h_{1:t}; r_{t+1} \mid q_{t+1} = q^\dagger)$ as Proposition 1 requires. □

In teaching terms, one early diagnostic response can determine which of two misconception branches applies to a learner, and subsequent interactions reveal nothing further about the branch. A log records what the learner has done; only a state represents the branch implied by that history.

B.3 Proposition 3

Proof. Take two concepts c_1, c_2 with exercise families $\mathcal{A}$ on c_1 and $\mathcal{B}$ on c_2. The true learner has mastered c_1 and not c_2, so a plan raises the probe score by $g > 0$ times the share of its exercises drawn from $\mathcal{B}$. True response probabilities are $P(r = 1 \mid \mathcal{A}) = \frac{1}{2} + \frac{\delta}{2}$ and $P(r = 1 \mid \mathcal{B}) = \frac{1}{2} - \frac{\delta}{2}$.

Let the memory hold the mirrored belief that c_2 is mastered and c_1 is not. It then assigns the same two response probabilities with the labels exchanged, so its per-step response-model error is $|(\frac{1}{2} + \frac{\delta}{2}) - (\frac{1}{2} - \frac{\delta}{2})| = \delta$ on either family. The memory imagines the mirrored gain, g times the share

of $\mathcal{A}$ exercises. A plan of length H maximizing the imagined gain therefore consists of $\mathcal{A}$ exercises only and achieves true gain 0. A policy that ignores IMAGINE and draws each of the H slots from $\mathcal{A}$ or $\mathcal{B}$ with equal probability achieves expected true gain $g/2 > 0$. The planner is worse by $g/2$ for every δ, in particular for every $\delta < \eta$. □

The construction separates the two error scales that the simulation bound $C(H)(\varepsilon + H\delta)$ combines. Response-model error can be made arbitrarily small while the *ordering* of imagined gains is inverted, because a teaching policy ranks candidates by learning gain, which is not the quantity a response model is fitted to. This separation is why imagination quality is measured on its own in Section 5, not read off predictive accuracy. It is also why a memory good enough to read from is not automatically good enough to plan against.

C Protocol details and judge disagreement

How each memory predicts. Every row of Table 1 comes from one evaluator, which asks a memory for the probability that the next exercise is answered correctly and scores that probability. Each memory answers with the estimator supported by its content. The world-model memory uses its prediction head, and BKT-in-prompt uses its per-concept posterior. Full context and retrieval use the mean accuracy of the learner over the concepts the exercise carries. They fall back to overall accuracy, and to one half before any interaction is written. No language model stands between a memory and its prediction. A difference between rows therefore reflects the memories rather than how well each is prompted.

The full context read by an LLM. Figure 5 scores full context with the log statistics it shares with retrieval. To test whether a language model gets more from the same text, each provider of Appendix F reads the first 50 interactions in the full-context serialization of Table 3, together with the identifiers and concepts of the next 100 exercises, and returns in one call the probability that each is answered correctly. The sample is the first window with 150 valid interactions of each of the first 100 test learners that have one, or of all 74 such learners on ASSISTments 2009, and the evaluator of Table 1 scores both LLMs, the world-model memory of seed 42, BKT-in-prompt and retrieval on these windows. All 548 replies are usable. Provider A does no better than retrieval on any corpus, and Provider B does better on the two ASSISTments corpora but not on NIPS34. Both stay below the world-model memory on every corpus, by 9.7, 14.3 and 15.2 points of rollout AUC for Provider A and by 8.6, 8.0 and 16.0 for Provider B on ASSISTments 2009, ASSISTments 2017 and NIPS34 (Table 4).

Table 4: The full context read by the two LLM providers against the memories of Table 1, scored on the same windows, the first window with 150 valid interactions of each of the first 100 test learners that have one (all 74 on ASSISTments 2009). The world-model memory is the model trained with seed 42

	ASSIST2009		ASSIST2017		NIPS34	
Memory	$\text{AUC}_{h=100}$	$r_{h=100}$	$\text{AUC}_{h=100}$	$r_{h=100}$	$\text{AUC}_{h=100}$	$r_{h=100}$
Provider A, full context	0.601	0.661	0.580	0.470	0.633	0.712
Provider B, full context	0.612	0.721	0.643	0.455	0.626	0.761
Mental world model (ours)	0.699	0.716	0.723	0.717	0.786	0.877
BKT-in-prompt	0.623	0.217	0.576	0.193	0.603	0.633
Retrieval	0.604	0.718	0.594	0.615	0.673	0.793

Exercises and the difficulty-matched control. Every model that reads an exercise, whether memory, simulated learner or judge, receives all the concepts the exercise carries, as in training. A policy targets one concept per exercise, the listed concept that tags the fewest exercises. This is the only concept on ASSISTments 2017 and the leaf of the subject hierarchy on NIPS34, whose exercises all list the same root first. The difficulty-matched control draws exercises at random from those whose accuracy lies within 0.08 of a target near the mean accuracy of the exercises the greedy policy teaches. The targets are 0.35, 0.70 and 0.72 on ASSISTments 2017, ASSISTments 2009 and NIPS34, where the greedy policy teaches at 0.344, 0.697 and 0.722.

Table 5: The policies of Table 2 and the best focus level on ASSISTments 2017 under every judge, as Section 4 commits to. Paired learner-level Δ gain against random sequencing. The world-model planner is the only policy in this table that stays positive and significant under all three judges

	Probe of the learner	Same-arch. judge	Cross-arch. judge	n
Difficulty-matched control	-0.042***	-0.026***	-0.041***	162
Greedy on weakest concept	+0.001	-0.028***	-0.010*	162
Focus-2 (best scripted)	+0.019***	+0.016*	-0.048***	162
Planner on BKT-in-prompt	-0.011*	+0.016***	+0.002	162
Planner on world-model memory	+0.013**	+0.016***	+0.013**	162
LLM tutor	+0.037*	+0.035	+0.009	22
LLM tutor, second provider	+0.063**	+0.016	-0.004	22

Table 6: The same split by judge for the two remaining corpora, restricted to conditions run on all three corpora

	Probe of the learner	Same-arch. judge	Cross-arch. judge	n
ASSIST2009				
Difficulty-matched control	+0.002	+0.009	+0.010	167
Greedy on weakest concept	+0.019*	+0.057***	+0.025**	167
Planner on BKT-in-prompt	-0.008	-0.002	-0.003	167
Planner on world-model memory	-0.004	+0.011	+0.005	167
NIPS34				
Difficulty-matched control	+0.007	+0.020***	+0.023***	273
Greedy on weakest concept	+0.096***	+0.057***	+0.079***	273
Planner on BKT-in-prompt	+0.006	+0.013*	+0.009	273
Planner on world-model memory	+0.011	+0.017**	+0.012*	273

Runs, seeds and tests. Every E2 and E3 cell pools three repetitions of the protocol (seeds 42–44) over the same test learners. On ASSISTments 2017, the planner cells on both memories and the random-sequencing cell are the exceptions and pool five repetitions (seeds 42–46). A comparison uses the seeds its two cells share. The second LLM provider, the eight-batch run of Section 5 and the memory comparison of Appendix F run on one seed. A seed repeats the tutoring loop, not training, and each trained model stays fixed across conditions. Each learner-level comparison is a one-sample t-test on per-learner differences against random sequencing unless another cell is named. A learner appearing under several seeds contributes the mean of its differences, so n counts learners rather than learner–seed pairs. The significance marks are $^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$, uncorrected.

The three judges. Section 4 commits to reporting the judges separately rather than averaging them, and this appendix does so for every policy of Table 2 and the best focus level. The first judge is the probe change of the simulated learner itself, the primary measurement. Two external judges predict correctness on the same held-out probe items from the same recorded sessions. The *same-architecture* judge is a world-model memory trained on a different seed, and the *cross-architecture* judge is an attentive knowledge-tracing model (Ghosh et al., 2020). Neither judge is the memory the tutor reads from. An effect that comes from exploiting the memory rather than from teaching the learner is expected to appear under a judge that shares the errors of that memory and to fade under one that does not.

Table 5 shows the cost of relying on a single measuring instrument. Two results survive the change of judge. The difficulty-matched control is negative under all three, and the world-model planner is the only policy in this table that is positive and significant under all three. The personalization effect of Section 5 holds under the cross-architecture judge ($+0.030$, $p < 10^{-6}$) but not under the same-architecture judge (-0.002, $p = 0.70$). The advantage of the best scripted focus level is positive under the probe of the simulated learner and the same-architecture judge but *negative* under the cross-architecture judge (-0.048). The advantage of the LLM tutor loses significance under both external judges ($+0.035$ and $+0.009$, with $+0.016$ and -0.004 for the second provider). On the other two corpora, greedy on the weakest concept is positive and significant under all three judges and stays ahead of the difficulty-matched control under all three (Table 6). The two planners do

Table 7: The same planner on two memories with faithful rollouts and one without, where WM is the world-model memory. Learner-level paired differences on ASSISTments 2017 under all three judges, pooled over seeds 42–44 for contrasts with the DKT planner and over seeds 42–46 for the other two; * marks $p < 0.05$ and ** marks $p < 0.01$

Contrast	Probe of the learner	Same-arch. judge	Cross-arch. judge
WM planner − DKT planner	-0.0254^{**}	-0.0409^{**}	-0.0290^{**}
DKT planner − BKT planner	$+0.0429^{**}$	$+0.0392^{**}$	$+0.0395^{**}$
WM planner − BKT planner	$+0.0249^{**}$	-0.0001	$+0.0108^{**}$
DKT planner − random	$+0.0308^{**}$	$+0.0532^{**}$	$+0.0364^{**}$
WM planner − random	$+0.0135^{**}$	$+0.0158^{**}$	$+0.0126^{**}$

not separate there under any judge. On ASSISTments 2017, the only teaching effect in Table 5 that survives every judge is that of the planner running IMAGINE on the world-model memory.

This paper takes the conservative reading. Consider a policy whose advantage appears only under judges that share an architecture with the memory it plans against. It has not been shown to teach; it has been shown to move a particular model. Reporting the judges separately makes that distinction visible, and averaging them hides exactly the policies for which it matters. The world-model planner is not among them because its advantage also holds under the cross-architecture judge.

A planner on a second faithful memory. In Section 5 the planner runs on the world-model memory and on BKT-in-prompt, a readable memory of low rollout fidelity. On ASSISTments 2017, Table 1 places DKT close to the world-model memory on the same rollout measure (0.724 against 0.741), although DKT exposes no interface a tutor can read. A mastery readout supplies such an interface. It asks the model for the probability that an exercise carrying a single concept and no item identity is answered correctly. With this readout, DKT also becomes a memory of the kind Section 3 proposes, a learned learner state that a policy reads and rolls forward. The same random-shooting planner then runs on either memory. Table 7 reports the outcome under the protocol above for the same simulated learners. Under the probe of the simulated learner, both faithful memories beat random sequencing, whereas BKT-in-prompt falls below it. Between the two, DKT gains 0.013 more at the learner level over seeds 42–51 under the probe of the simulated learner, but the difference does not reach significance with seeds as the unit ($p = 0.10$). With seeds as the unit, the DKT planner also beats random sequencing ($+0.025$, $p = 0.004$) and BKT-in-prompt in all ten seeds ($+0.034$, $p < 0.001$). Under that probe, rollout fidelity thus separates the two memories above random sequencing from the one below it, but it does not order the two faithful ones. Proposition 3 leads one to expect this, since a planner ranks candidates by learning gain, which is not the quantity a response model is fitted to.

Seeds as the unit of analysis. The preceding tests average each learner over seeds and test across learners. Using the protocol seed as the unit instead leaves few degrees of freedom and little power, but shows whether a sign is stable. The seed is also the unit at which the planner samples, since the 32 candidate sessions it draws at each session start depend only on the protocol seed and are scored for every learner of that seed. Table 8 reports both analyses for every seed from 42 to 46 available to each contrast. The two units agree on all three point estimates. Personalization and imagination quality reach significance under either unit. The margin of the planner over random sequencing is positive in four of five seeds but does not reach significance at that count. Five more protocol seeds (47–51) rerun the three planner cells with every other setting unchanged, and a rerun of seed 42 reproduces the recorded runs exactly. Over the ten seeds, the planner on the world-model memory beats random sequencing in eight ($+0.012$, $p = 0.037$ with seeds as the unit) and gains more than on BKT-in-prompt in all ten ($+0.021$, $p = 0.001$). The planner on BKT-in-prompt falls below random sequencing in eight (-0.009, $p = 0.012$). At the learner level, the margin of the world-model planner over random sequencing is $+0.014$ under the same-architecture judge and $+0.012$ under the cross-architecture judge, both with $p < 10^{-4}$.

A second simulated learner. Every tutoring run above simulates learners with the AKT model trained with seed 42. The planner contrasts of Section 5.2 are rerun for seeds 42–46 with the DKT model trained with seed 42 as the learner and every other setting unchanged (Table 9). The rerun

Table 8: Every E2 headline on ASSISTments 2017 under both units of analysis, measured by the probe of the simulated learner. The learner level follows the protocol of this appendix. The seed level treats each protocol seed as one observation; * marks $p < 0.05$ and ** marks $p < 0.01$

Claim	Learner level	Seed level	Seeds	Per seed
Personalization	$+0.0429^{**}$	$+0.0429^{**}$	3	+0.041, +0.051, +0.037
Imagination quality	$+0.0249^{**}$	$+0.0249^{*}$	5	+0.013, +0.018, +0.022, +0.053, +0.019
Planning against random	$+0.0135^{**}$	$+0.0135$	5	-0.007, +0.020, +0.003, +0.041, +0.010

keeps the 162 test learners, their windows, the probe items and the two judges. Practice moves the DKT learner less than the AKT learner. At the learner level, its probe gains 0.008 under random sequencing, against 0.054 for the AKT learner. Under the planner, the probe gains 0.015 on the world-model memory and 0.004 on BKT-in-prompt. Neither planner separates from random sequencing under this probe, and all three contrasts are smaller in magnitude than with the AKT learner. The planner still gains 0.011 more on the world-model memory than on BKT-in-prompt ($p = 0.024$), and the difference is positive in three of five seeds (seed level $p = 0.26$). The two judges place this difference within noise (-0.002 and $+0.003$). The difference between the memories thus keeps its direction at the learner level under the probe of a second simulated learner.

Table 9: The planner contrasts of Section 5.2 with the DKT model trained with seed 42 as the simulated learner in place of AKT, ASSISTments 2017, seeds 42–46. Learner-level paired differences under the probe of the simulated learner and the two external judges of Table 5. The last column counts the seeds with a positive difference under the probe; * marks a learner-level $p < 0.05$

Contrast	Probe of the learner	Same-arch. judge	Cross-arch. judge	Seeds > 0
WM planner − BKT planner	$+0.0112^{*}$	-0.0022	$+0.0028$	3 of 5
WM planner − random	$+0.0064$	$+0.0074$	-0.0003	2 of 5
BKT planner − random	-0.0048	$+0.0096^{*}$	-0.0031	1 of 5

D ROLLOUT-OBJECTIVE ABLATION

Rollout fidelity responds more to the training objective than to capacity. Every configuration below is ranked by *validation* rollout performance on seed 42, and each selected configuration is then scored on the test split over seeds 42–44.

Table 10 reports a sweep of single changes and their combinations on ASSISTments 2017, seed 42. Each configuration is scored by mean rollout AUC over $h \in \{20, 50, 100\}$ on the validation split. The search runs in two passes. The first (upper block) perturbs the reported configuration, and the second (lower block) perturbs the depth and weight that the first selects. The second pass also keeps the checkpoint with the best validation rollout AUC rather than single-step AUC. It scores each checkpoint on one sampled trajectory of 20 steps after 50 observed interactions. Against the reported configuration, latent width is worth $+0.0005$ and hidden width is worth $+0.0035$, so capacity does not separate the configurations at that setting. Doubling the depth of the overshoot term is worth $+0.0038$, and doubling its weight is worth $+0.0068$. Applying both changes with a latent width of 64 gives $+0.0146$, more than the sum of the three separate gains. Carrying both further, to $d = 15$ and $\lambda = 0.9$, gives $+0.0213$.

From there, a lighter reconstruction weight, a halved learning rate and lighter dropout are each worth between $+0.0014$ and $+0.0025$. All three together are worth $+0.0035$. The search then tries a heavier KL weight and a wider hidden layer on top of them. Both changes reduce rollout AUC, so the search stops.

The selected configuration is retrained on seeds 43 and 44, and Table 11 reports all three seeds on the test split, with teacher-forced predictions beside closed-loop rollouts. Rollout AUC rises at every horizon. The rise grows with horizon, from $+0.0102$ at $h = 5$ to $+0.0307$ at $h = 100$. A paired t-test over the three seeds gives $p = 0.015$ at $h = 5$ and $p = 0.002$ at $h = 100$. Per-learner correlation also rises, $+0.033$ at $h = 20$ ($p = 0.024$) and $+0.045$ at $h = 100$ ($p = 0.004$).

Table 10: Rollout-objective sweep on the ASSISTments 2017 validation split, seed 42. Each indented row changes the factors it names from the row it is indented under. The value in parentheses is the change in the mean from the reported configuration. Within the sweep, the overshoot term, not capacity, separates the configurations. The lower block also keeps the checkpoint with the best validation rollout AUC, and bold marks the selected configuration

Configuration	h=20	h=50	h=100	Mean
reported (d=5, λ=0.3, $\lvert z\rvert$=32)	0.7456	0.7430	0.7404	0.7430
$\lvert z\rvert$=64	0.7412	0.7466	0.7429	0.7435 (+0.0005)
β_{KL}=0.05	0.7462	0.7454	0.7430	0.7449 (+0.0019)
width 320	0.7470	0.7476	0.7451	0.7466 (+0.0035)
d=10	0.7473	0.7477	0.7456	0.7469 (+0.0038)
λ=0.6	0.7485	0.7520	0.7491	0.7498 (+0.0068)
d=10, λ=0.6, $\lvert z\rvert$=64	0.7566	0.7593	0.7570	0.7576 (+0.0146)
d=15, λ=0.9, $\lvert z\rvert$=64	0.7606	0.7662	0.7661	0.7643 (+0.0213)
dropout 0.1	0.7609	0.7683	0.7679	0.7657 (+0.0227)
β_{KL}=0.20	0.7618	0.7680	0.7669	0.7656 (+0.0225)
lr 5×10^{-4}	0.7632	0.7687	0.7680	0.7666 (+0.0236)
λ_{rec}=0.1	0.7629	0.7695	0.7679	0.7668 (+0.0237)
all three together	0.7625	0.7707	0.7702	0.7678 (+0.0247)
+ β_{KL}=0.20	0.7627	0.7693	0.7691	0.7670 (+0.0240)
+ β_{KL}=0.20, width 320	0.7614	0.7681	0.7678	0.7658 (+0.0227)

Table 11: The selected configuration against the reported one on the test split, mean over seeds 42–44. The tuning moves the rollout and the teacher-forced readings of one checkpoint in opposite directions

	Teacher-forced AUC		Rollout AUC		
Horizon	Reported	Selected	Reported	Selected	Δ rollout
h=5	0.7865	0.7832	0.7697	0.7799	+0.0102
h=10	0.7829	0.7781	0.7591	0.7746	+0.0155
h=20	0.7816	0.7752	0.7510	0.7711	+0.0201
h=50	0.7813	0.7754	0.7434	0.7702	+0.0268
h=100	0.7847	0.7786	0.7414	0.7722	+0.0307

Teacher-forced AUC falls, -0.0064 at $h = 20$ ($p = 0.029$) and -0.0060 at $h = 100$ ($p = 0.017$). The two readings of one checkpoint move in opposite directions at every horizon and in every seed. A configuration that simply learns to predict responses better raises both. The selected configuration improves the reading that a tutoring agent uses, at the cost of teacher-forced AUC. That reading comes from rolling the memory forward under its own predictions. The calibration error rises by 0.019 at $h = 100$ ($p = 0.09$). The mean absolute error of per-learner accuracy changes by -0.006 at $h = 100$ ($p = 0.24$).

Removing the overshoot term separates its share of the gain from that of the checkpoint rule. The ablation retrains the setting that heads the lower block of Table 10 with seeds 42–44 and the overshoot weight at zero, under the same checkpoint rule. The unablated setting is the first row of Table 13. Rollout AUC at $h = 100$ on the test split falls from 0.769 to 0.722, below the reported configuration of Table 1. The loss of 0.047 ($p = 0.002$ on a paired test over seeds) occurs in each seed, so the checkpoint rule alone does not account for the gain.

Each corpus is swept on its validation split rather than inheriting the ASSISTments 2017 setting, because one corpus cannot show how far the objective transfers across corpora. Table 12 reports each selected configuration retrained with three seeds. ASSISTments 2017 gains $+0.031$ ($p = 0.002$), which puts the memory 0.048 above the strongest knowledge-tracing model in Table 1. ASSISTments 2009 gains $+0.005$ and does not separate from zero ($p = 0.53$). Two seeds rise, and the third falls by 0.008. NIPS34 gains $+0.0006$. The eight configurations swept there span 0.0011 on validation, the pattern of a search over a flat surface rather than of a setting not yet found.

Table 12: Tuning the training on each corpus separately, test split at $h = 100$, mean $\pm$ standard deviation over seeds 42–44. Each configuration wins the validation sweep of its corpus except on NIPS34, where it ties with $d = 25$, $\lambda = 0.9$ at four decimals. Every tuned configuration uses $|z| = 64$. Only the ASSISTments 2017 configuration separates from the reported one over three seeds; ** marks $p < 0.01$ on a paired test

Dataset	Tuned configuration	Reported	Tuned	Δ
ASSIST2009	d=15, λ=0.6	0.724 $\pm$0.006	0.730 $\pm$0.004	$+0.0052$
ASSIST2017	d=15, λ=0.9, rec/lr/dropout	0.741 $\pm$0.002	0.772 $\pm$0.000	$+0.0307^{**}$
NIPS34	d=20, λ=0.9	0.774 $\pm$0.000	0.775 $\pm$0.000	$+0.0006$

Table 13: Design-switch ablations on the ASSISTments 2017 test split at $h = 100$, mean $\pm$ standard deviation over seeds 42–44. Each switch is turned off; * marks $p < 0.05$ and ** marks $p < 0.01$ on a paired test over seeds against the first row

Switch	Variant	$\text{AUC}_{h=100}$	Δ
—	Full design	0.769 $\pm$0.001	—
`bind_interaction`	exercise and response enter the cell separately	0.763 $\pm$0.000	-0.0063^{*}
`share_decoder`	one head for prediction and reconstruction	0.702 $\pm$0.004	-0.0677^{**}

The best setting differs by corpus. On both ASSISTments corpora, overshoot depth 15 is best and depth 20 scores lower, but the best weight is 0.6 on 2009 and 0.9 on 2017. Applying the 2017 setting to 2009 recovers $+0.011$ of the $+0.014$ that the 2009 sweep reaches on validation. The three extra factors also do not transfer across corpora. On top of the depth and weight selected for each corpus, they leave ASSISTments 2009 at $+0.0023$ against $+0.0140$ without them, and leave NIPS34 at $+0.0007$ against $+0.0011$. On all three corpora, the gain grows with rollout depth, as in the single-corpus ablation. At $h = 5$ it is slightly negative on ASSISTments 2009 and NIPS34.

The main results are obtained without any of this tuning. The tutoring experiments of the main text use the memory configuration in Table 1, so none of their results depends on this tuning.

On ASSISTments 2017, the planner of Section 5.2 also runs on the seed-42 model that heads the lower block of Table 10. The AKT learner and seeds 42–46 are those of Section 5.2. This memory reaches a rollout AUC of 0.770 at $h = 100$, against 0.739 for the reported memory with the same seed. At the learner level, the planner gains 0.006 less on this memory than on the reported one ($p = 0.10$), and this memory leads in two of five seeds. Neither judge separates the two memories. On this memory, the planner stays 0.019 ahead of the planner on BKT-in-prompt ($p < 10^{-5}$) and leads it in each of five seeds. Its margin of 0.008 over random sequencing does not reach significance ($p = 0.062$). The higher rollout AUC of this memory thus does not raise the gain of the planner, as with the two faithful memories of Appendix C.

E Design ablation

Two choices in the write path are part of the design rather than hyperparameters. The exercise and response enter the recurrent cell as one bound interaction rather than as two inputs. Prediction and reconstruction use separate heads rather than one. Each choice is ablated in Table 13 (ASSISTments 2017, the setting that heads the lower block of Table 10, three seeds, test split).

Turning either off costs more than any change of capacity in the search of Appendix D. Unbinding the interaction costs 0.006 ($p = 0.013$), with a loss in each of the three seeds. Sharing one head costs 0.068 ($p = 0.003$) and puts the memory below both knowledge-tracing models in Table 1. This cost exceeds twice the 0.031 gain from tuning the training (Table 12). A single head has to use the same parameters to represent the response it reconstructs and the response it predicts. These quantities differ because one is observed and the other is a belief about what has not happened yet.

F LLM TUTOR PROMPTS AND FALLBACK ACCOUNTING

What the tutor sees. In the read-only condition, the serialization $R_\phi(b_t)$ of Section 3 is the only channel through which the LLM tutor sees the learner. For the world-model memory, the serialization contains overall mastery and then the 12 weakest concepts with mastery, uncertainty and recent trend. Each block covers one learner, in this format:

```
Overall mastery: <mean mastery>
Weakest concepts:
  concept_<id>: <mastery> (±<uncertainty>), trend <change>
  concept_<id>: <mastery> (±<uncertainty>), trend <change>
  ...
```

At 12 concepts this block costs 213 tokens by the count of Section 5, against 94 for the five-concept readout measured there.

Planning prompt. One call plans one session. Braced names are slots filled at each call.

```
You are a tutor planning the next practice session for one student.

STUDENT MEMORY (from the tutoring system's learner model):
{readout}

AVAILABLE EXERCISES (one representative per concept; difficulty = fraction
of students answering correctly):
{menu}
{imagined}
Plan a session of exactly {k} exercises for this student (repetition allowed).
Reply with JSON only, no other text: {"session": [item_id, item_id, ...]}
```

The difficulty values behind the exercise menu are computed once per protocol seed as the accuracy of each exercise over all attempts in the first evaluation batch. Each menu line shows one minus this accuracy as the share of wrong answers, for example `concept_1: exercise 2104 (difficulty 67% answer wrong)`. These values enter only this menu and the difficulty-matched control. In the read-only condition, the {`imagined`} slot is empty. In the imagined-candidate condition, it carries the predicted mastery gain of each of three heuristic candidate sessions, averaged over four rollouts:

```
IMAGINED OUTCOMES (the learner model simulated these candidate sessions):
{lines}
You may pick one candidate, modify it, or plan a different session entirely.
```

The self-revision condition instead rolls out the plan the tutor has just drafted and returns it for revision, which nominally costs two calls per session rather than one.

Fallback accounting. Provider A is called through the Claude Code client with the model alias haiku and Provider B through the Codex client with its default model, both with default settings otherwise, and the clients resolve them to `claude-haiku-4-5-20251001` and `gpt-5.6-sol`. A plan is usable if it is JSON naming schedulable item ids. A reply that parses but names other ids goes back to the tutor once with the rejected plan appended. Greedy on the weakest concept plans the session when a call yields no usable plan, because the reply does not parse, the second answer is still unusable or the command-line client returns nothing after three attempts. In the self-revision

Table 14: Sessions planned by the greedy fallback per condition. No condition falls back in any session, over 4888 calls and the 2364 calls of the eight-batch run

Provider	Prompt	Seeds	Calls	Fallbacks
Provider A	read-only serialization	3	916	0
	the same prompt, eight batches	1	2364	0
	+ imagined candidates	3	917	0
	+ imagined self-revision	3	1834	0
Provider B	read-only serialization	1	305	0
	+ imagined candidates	1	305	0
	+ imagined self-revision	1	611	0

condition, a failed revision keeps the draft. No session is thus left unplanned. Table 14 counts the sessions that the fallback plans.

The memory behind the channel. Holding the prompts above fixed makes the memory the only thing that varies, so the same tutor can run again on a different memory. With Provider B, one seed and one batch, the tutor teaches the same 22 learners twice, once reading the world-model memory and once reading BKT-in-prompt. The protocol is built on the probe of the simulated learner, and this probe puts the world-model memory ahead by $+0.091$ ($p = 10^{-4}$). The difference is $+0.030$ ($p = 0.22$) under the same-architecture judge and $+0.005$ ($p = 0.83$) under the cross-architecture judge. Both arms issue 305 calls and plan every session without the fallback.