Spaced Interleaving Optimization via Curriculum Reinforcement Learning: A Neuro-Symbolic POMDP Approach

1. Executive Summary

The optimization of instructional sequencing—specifically the dynamic spacing and interleaving of practice items across multiple concepts—represents the "holy grail" of Intelligent Tutoring Systems (ITS). While the pedagogical benefits of **spaced repetition** (scheduling reviews to mitigate forgetting) and **interleaving** (mixing different problem types to improve discrimination) are well-established in cognitive science, implementing these strategies in automated systems remains computationally intractable under traditional heuristic frameworks. Standard approaches, such as fixed-interval schedulers or simple **Mastery Learning** thresholds (e.g., stopping practice at 75% accuracy), fail to account for the complex, high-dimensional temporal dynamics of human memory. They optimize for short-term fluency rather than long-term retention, leading to the inefficient "over-practice" of mastered skills and the "under-practice" of decaying ones.

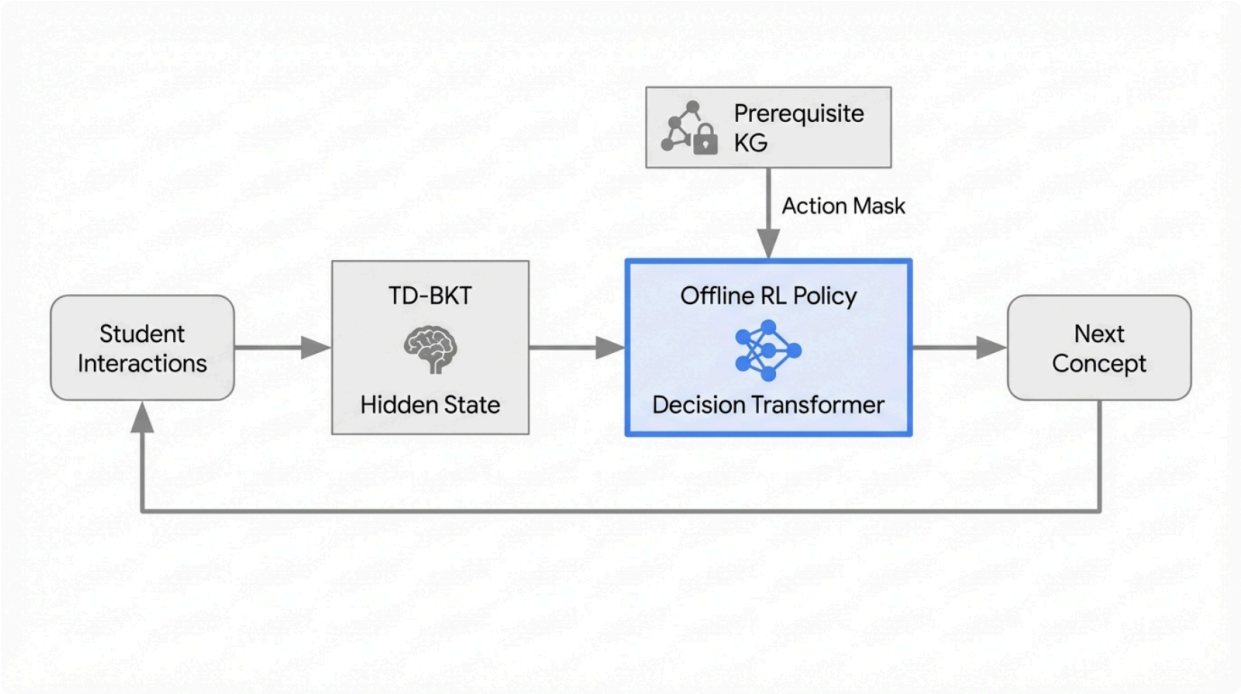
This report presents a comprehensive theoretical and technical framework for a **Curriculum Reinforcement Learning (CRL)** system designed to solve the spaced interleaving problem. We formalize the learning process as a **Partially Observable Markov Decision Process (POMDP)**, acknowledging that a student's true cognitive state is latent and must be inferred from noisy interaction logs. To tackle the state estimation challenge, we integrate **Temporal Difference Bayesian Knowledge Tracing (TD-BKT)**, an advanced probabilistic model that incorporates temporal difference information into belief state estimation, offering a superior signal to static BKT.

Central to this architecture is the application of **Offline Reinforcement Learning**. Recognizing the high ethical risks and logistical constraints of training RL agents directly on human students ("online" RL), we employ offline algorithms—specifically **Conservative Q-Learning (CQL)** and **Decision Transformers (DT)**—to extract optimal pedagogical policies from historical student trajectory datasets. These algorithms allow the system to learn "super-human" teaching strategies by identifying and "stitching" together optimal subsequences from sub-optimal historical data, effectively turning the "exploration-exploitation" dilemma into a data-mining advantage.

Furthermore, the system enforces pedagogical coherence through **Knowledge Graph (KG) Action Masking**, ensuring that the RL agent respects prerequisite constraints, and optimizes a novel reward function based on **Long-Term Retention** proxies (specifically **Half-Life Regression**) rather than immediate binary correctness. This report provides an exhaustive

blueprint for implementing this neuro-symbolic system, benchmarking it against standard 0.75 mastery threshold baselines, and analyzing its potential to revolutionize personalized curriculum sequencing.

Architecture of the Spaced Interleaving Optimization System



The system architecture acts as a closed-loop control system. Student interactions are processed by the TD-BKT module to update the Belief State. This state is fed into the Offline RL Policy (Decision Transformer), which selects the next optimal concept to practice. The selection is constrained by a dynamic Action Mask derived from the Prerequisite Knowledge Graph, ensuring pedagogical validity.

---

2. Theoretical Formulation: The Learning POMDP

The fundamental challenge of personalized education is decision-making under uncertainty. A tutor (human or machine) never has direct access to a student's brain (the *state*). They can only observe the student's outputs (answers to questions, time taken, hints requested). This partially observable nature makes the **Partially Observable Markov Decision Process (POMDP)** the theoretically ideal framework for modeling curriculum sequencing.[1] In a standard Markov Decision Process (MDP), the agent knows the state of the world. In education, assuming the state is known leads to "cognitive diagnosis deviation" [4], where the

system misinterprets a lucky guess as mastery or a careless slip as ignorance.

In this context, we define the spaced interleaving problem not merely as a scheduling task, but as a sequential optimization problem where the agent must decide *which* concept to present next to maximize the student's future retention. This requires balancing the need to reinforce weak concepts (spacing) with the need to introduce new ones (interleaving), all while navigating the constraints of the student's current zone of proximal development.

2.1 The Curse of Dimensionality in Educational State Spaces

Before formalizing the tuple, it is crucial to understand the scale of the problem. If a curriculum has K concepts (e.g., 100 skills in a math course), and each concept has a mastery level $m \in $ and a memory strength $d \in \mathbb{R}^+$ (decay rate), the state space is continuous and $2K$ -dimensional.

$$\mathcal{S} \subset^K \times \mathbb{R}^K$$

For just 20 skills, this space is effectively infinite. Traditional tabular RL methods (like Q-tables) fail immediately here. This necessitates the use of **Function Approximation** (via Deep Neural Networks in CQL or Decision Transformers) to generalize across this vast state space.

2.2 Formal Definition of the POMDP Tuple

We formally define the learning process as a tuple $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{T}, \mathcal{Z}, \mathcal{R}, \gamma \rangle$:

State Space ($\mathcal{S}$)

The latent cognitive state of the student at time t, denoted as s_t. As distinct from simple BKT models that track binary mastery, our CRL system requires a richer state representation to support spacing optimization.

- **Mastery Vector (L_t):** A vector of size K representing the probability of mastery for each concept k. $L_t \in^K$.
- **Recency/Decay Vector (Δ_t):** A vector tracking the time elapsed since the last interaction with each concept. This is critical for modeling the "forgetting curve" dynamics.
- **Skill Parameters (Θ):** Latent traits such as "learning rate" or "initial knowledge," often inferred from static embeddings or historical priors.

- **Full State:** $s_t =$.

Action Space ($\mathcal{A}$)

The set of all possible pedagogical activities. In an interleaving context, an action a_t corresponds to selecting a specific concept k to practice.

- **Discrete Formulation:** $\mathcal{A} = \{1, 2, ..., K\}$. The agent selects the index of the next problem type.
- **Interleaving Constraint:** Unlike "blocked" practice (doing A, A, A, then B, B, B), the agent can choose any $a_t \in \mathcal{A}$ at any step, provided prerequisites are met.

Observation Space ($\mathcal{O}$)

The observable response from the student after taking action a_t. This is the noisy lens through which we view s_t.

- **Correctness (y_t):** $y_t \in \{0, 1\}$ (Binary) or $y_t \in$ (Partial Credit).
- **Response Time (τ_t):** $\tau_t \in \mathbb{R}^+$. A very fast correct answer implies high mastery; a slow correct answer implies struggle.
- **Observation Tuple:** $o_t = \langle y_t, \tau_t \rangle$.

Transition Function ($\mathcal{T}$)

$\mathcal{T}(s'|s, a) = P(s_{t+1}|s_t, a_t)$. This function models the *learning process* itself.

- It encompasses the probability that practicing concept a_t transitions the student from "Unmastered" to "Mastered" (Learning).
- Crucially for our system, it also models the **Forgetting Process**: even for actions *not* taken ($a \neq k$), the state component Δ_t increases, and the mastery probability L_t may decrease according to the decay curve. This "passive transition" is what makes the spacing problem dynamic and difficult.

Observation Function ($\mathcal{Z}$)

$\mathcal{Z}(o|s, a) = P(o_t|s_t, a_t)$. The probability of observing outcome o given latent state s.

- This captures **Slip** probabilities (Mastered state $\rightarrow$ Incorrect answer) and **Guess** probabilities (Unmastered state $\rightarrow$ Correct answer).
- Standard BKT assumes these are static constants. Our system, using TD-BKT, allows these to be influenced by temporal factors (e.g., slipping is more likely if Δ_t is high).

Reward Function ($\mathcal{R}$)

The objective function that drives the RL agent.

- **Naive Reward:** $r_t = y_t$ (Immediate correctness). *Critique:* This leads to "easy" policies where the agent asks questions the student already knows, maximizing immediate reward but failing to teach.
- **Curriculum Reward:** r_t must reflect **Long-Term Retention**. We define this mathematically in Section 9 using Half-Life Regression deltas.

Discount Factor (γ)

Determines the planning horizon. For spaced repetition, γ must be close to 1 (e.g., 0.99 or 0.995). The benefits of a spaced review today might not materialize until a retention test 30 days from now. A low γ (myopic agent) would fail to prioritize these long-term gains.

---

3. State Estimation: Temporal Difference Bayesian Knowledge Tracing (TD-BKT)

Since s_t is hidden, the RL agent cannot act on it directly. It must act on a **Belief State** b_t, which is a sufficient statistic or probability distribution over possible states $b_t(s) = P(s_t = s|h_t)$. In high-dimensional POMDPs, maintaining an exact belief state is computationally intractable. We therefore use **TD-BKT** as a structured, tractable *state estimator* to generate the input features for our RL policy.[4]

3.1 The Limitations of Standard BKT for Spacing

Standard Bayesian Knowledge Tracing (BKT) uses a Hidden Markov Model (HMM) structure. It updates the probability of mastery L_t based on a fixed learn rate $P(T)$ and the observation o_t:

$$P(L_t|L_{t-1}) = L_{t-1} + (1 - L_{t-1})P(T)$$

This equation has a fatal flaw for spacing optimization: it is **time-invariant**. It treats the interval between t and $t+1$ as irrelevant. Whether the next opportunity is 1 minute later or 1 month later, the model assumes the same probability of learning and *zero* probability of forgetting (unless explicitly modified with a complex "forgetting" parameter which is rarely used in standard implementations). This leads to "cognitive diagnosis deviation" [4], where the model fails to track the natural decay of memory over time.

3.2 TD-BKT: Incorporating Temporal Dynamics

TD-BKT [4] introduces "temporal difference information" into the update rule. It assumes that the cognitive state is subject to changes based on the time elapsed and the identification of "cognitive inflection points" (moments where learning or forgetting accelerates).

We implement the TD-BKT belief update as a two-step filter: **Time Update** (Prediction) and **Measurement Update** (Correction).

Step 1: The Prediction Step (Time & Decay)

Before seeing the observation at time t, we project the student's mastery forward based on the time elapsed $\tau = t - t_{last}$.

Standard BKT assumes $P(L_t) \approx P(L_{t-1})$. TD-BKT modifies this:

L_{t|t-1} = L_{t-1} \cdot \mathcal{F}(\tau)$$Where $\mathcal{F}(\tau)$ is a decay function derived from the **Ebbinghaus Forgetting Curve** or **Half-Life Regression (HLR)** models (see Section 9). A common formulation used in conjunction with BKT for spacing is:$$\mathcal{F}(\tau) = 2^{-\frac{\tau}{h}}

Here, h is the "half-life" or strength of the memory. This step explicitly models that *mastery probability decreases as time passes without practice*. This provides the RL agent with the crucial "urgency" signal needed to schedule a review.

Step 2: The Correction Step (Observation)

Once action a_t is taken and observation o_t (correct/incorrect) is received, we update the belief using Bayes' rule, but using the *decayed* prior $L_{t|t-1}$ instead of the previous posterior.

If $o_t = 1$ (Correct):

$$L_t = \frac{L_{t|t-1} \cdot (1 - P(S))}{L_{t|t-1} \cdot (1 - P(S)) + (1 - L_{t|t-1}) \cdot P(G)}$$

If $o_t = 0$ (Incorrect):

$$L_t = \frac{L_{t|t-1} \cdot P(S)}{L_{t|t-1} \cdot P(S) + (1 - L_{t|t-1}) \cdot (1 - P(G))}$$

Where $P(S)$ is the probability of a Slip and $P(G)$ is the probability of a Guess.

3.3 The Belief State Vector

The output of the TD-BKT module is a high-dimensional vector that serves as the input s_t for the RL agent. For a curriculum of K skills, the belief state vector b_t is:

$$b_t =$$

- $\hat{p}(L_t^k)$: The current estimated probability of mastery for skill k (0.0 to 1.0).
- Δ_t^k : The normalized time elapsed since skill k was last practiced.

This vector provides the RL agent with a complete "dashboard" of the student's mind: which concepts are mastered, which are decaying, and which are currently being learned.[1]

---

4. Constraint Satisfaction: Knowledge Graph Action Masking

A pure RL agent, if left unconstrained, might discover "degenerate" strategies. For example, it might find that "Interleaving Concept Z (Advanced Calculus) immediately after Concept A (Basic Addition)" maximizes some obscure reward signal in the dataset, even if the student lacks the prerequisites for Z. This results in an efficient but pedagogically invalid curriculum. To ensure the safety and coherence of the generated curriculum, we implement **Action Masking** based on a **Prerequisite Knowledge Graph**.[9]

4.1 Topology of the Prerequisite Graph

We model the curriculum structure as a Directed Acyclic Graph (DAG) $G = (V, E)$, where:

- V is the set of concepts (corresponding to the action space $\mathcal{A}$).

- E is the set of directed edges (u, v) where a directed edge implies that concept u is a strict prerequisite for concept v.

4.2 Dynamic Action Masking Mechanism

At every timestep t, the system computes a dynamic set of "valid" actions $\mathcal{A}_{valid}(t) \subseteq \mathcal{A}$. The logic is derived from the current Belief State b_t provided by the TD-BKT module.

An action (concept) v is considered valid if and only if:

1. **Prerequisites Met:** All immediate parents of v in the graph G have a mastery probability above a "prerequisite threshold" θ_{prereq} (e.g., 0.85).

$$\forall u \in Parent(v) : \hat{p}(L_t^u) \geq \theta_{prereq}$$

2. **Roots are Always Valid:** If v has no parents (root node), it is always valid (unless explicitly mastered and retired).

This creates a **Mask Vector** $M_t \in \{0, 1\}^K$:

$$M_t[k] = \begin{cases} 1 & \text{if } k \in \mathcal{A}_{valid}(t) \\ 0 & \text{otherwise} \end{cases}$$

4.3 Integration with Deep RL (Logits Masking)

In our Deep RL architectures (both CQL and Decision Transformer), the final layer of the neural network outputs a vector of **logits** z_t (unnormalized scores) for each possible action. Standard softmax would convert these to probabilities: $\pi(a) = \frac{e^{z_a}}{\sum e^{z_i}}$.

To strictly enforce the constraints, we apply the mask *before* the softmax operation using an additive penalty:

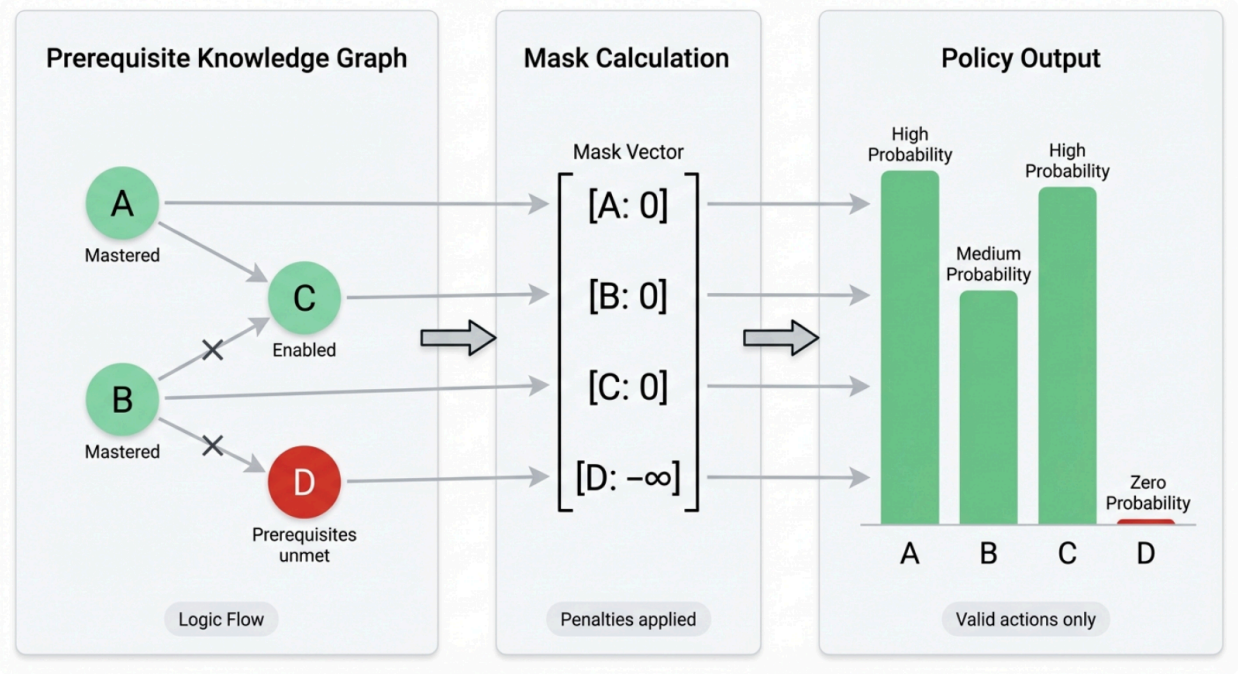
$$\tilde{z}_t = z_t + (1 - M_t) \cdot (-\infty)$$

$$\pi(a|s_t) = \text{Softmax}(\tilde{z}_t)$$

By adding $-\infty$ (or a very large negative number like -10^9) to the logits of invalid actions, their probability becomes effectively zero: $e^{-\infty} \approx 0$. This ensures that the RL agent

distributes 100% of its probability mass only among the pedagogically valid concepts.[12] This technique transforms the exploration process from "random chaos" to "structured discovery" within the student's Zone of Proximal Development.

Constraint Enforcement via Knowledge Graph Masking



The Dynamic Action Masking process. (Left) The Prerequisite Knowledge Graph shows Concept C requires Concept A. (Middle) The Belief State indicates Concept A is mastered (Green), enabling Concept C. Concept D's prerequisites are unmet (Red). (Right) The Mask Vector applies -Infinity penalties to invalid actions, forcing the Policy Distribution to select only from pedagogically valid concepts.

---

5. Offline Curriculum Reinforcement Learning

The engine of our system is the decision-making policy. We adopt an **Offline Reinforcement Learning** approach. In traditional "Online" RL, an agent learns by trial-and-error, interacting with the environment. In education, "trial-and-error" means experimenting on students with potentially confusing or suboptimal lesson sequences. This is ethically unacceptable and logistically expensive.

Offline RL allows us to learn a policy $\pi(a|s)$ entirely from a static dataset of historical

student interactions $\mathcal{D} = \{(s_t, a_t, r_t, s_{t+1})\}$, collected from previous iterations of the course or existing logs (e.g., EdNet, ASSISTments).[14] The goal is to learn a policy that outperforms the "behavioral policy" (the teacher or algorithm that generated the data).

We compare and implement two distinct Offline RL architectures: **Conservative Q-Learning (CQL)** and the **Decision Transformer (DT)**.

5.1 Algorithm I: Conservative Q-Learning (CQL)

CQL [17] is a value-based method that addresses the primary failure mode of offline RL:

Overestimation Bias. When an RL agent sees a state s in the dataset, it might estimate the value $Q(s, a)$ for an action a that was rarely (or never) taken in the history. Standard Q-learning tends to optimistically overestimate this value ("maybe this un-tried action is amazing!"). In offline settings, since the agent cannot try the action to verify, this overestimation propagates, leading to a policy that chooses "hallucinated" good actions that fail in reality.

CQL Formulation for Curriculum

CQL adds a conservative regularization term to the standard Bellman error objective. We seek to minimize the expected Q-value under the learned policy while maximizing the Q-value of the actual data samples.

The loss function for the Q-network is:

$$\mathcal{L}_{CQL}(\theta) = \alpha \left(\mathbb{E}_{s \sim \mathcal{D}} \left[\log \sum_{a} \exp(Q(s, a)) - \mathbb{E}_{a \sim \pi_\beta(s)} [Q(s, a)] \right] \right) + \frac{1}{2} \mathbb{E}_{s, a, s' \sim \mathcal{D}} \left$$

- **Term 1 (Minimization):** $\log \sum \exp(Q(s, a))$ pushes down the Q-values of all actions in state s.
- **Term 2 (Maximization):** $Q(s, a_{actual})$ pushes *up* the value of the action actually taken in the dataset.
- **Result:** The Q-values for actions *not* in the dataset (Out-of-Distribution or OOD actions) are pushed down relative to the data.

Why this matters for Spaced Interleaving:

If the historical data never shows a student practicing "Concept A" followed immediately by "Concept Z," CQL will assume this sequence is bad (low value) unless there is strong generalization evidence otherwise. This makes CQL a **Safe** algorithm for educational deployment. It ensures the agent stays "close" to the known valid pedagogical strategies

while optimizing within that support.

5.2 Algorithm II: Decision Transformer (DT)

The Decision Transformer [19] treats RL not as a dynamic programming problem (like Q-learning) but as a **Sequence Modeling** problem, leveraging the architecture of GPT-style Transformers.

Trajectory Representation

Instead of calculating value functions, we model the joint distribution of the trajectory. The input sequence to the Transformer is:

$$\tau = (\hat{R}_1, s_1, a_1, \hat{R}_2, s_2, a_2, ..., \hat{R}_T, s_T, a_T)$$

- $\hat{R}_t$ **(Return-to-Go):** The sum of future rewards desired from time t. This is the "prompt" for the model. For example, if we want high retention, we feed a high $\hat{R}_1$ value.
- s_t**:** The TD-BKT belief state embedding (continuous vector).
- a_t**:** The action token (discrete concept ID).

The "Stitching" Capability

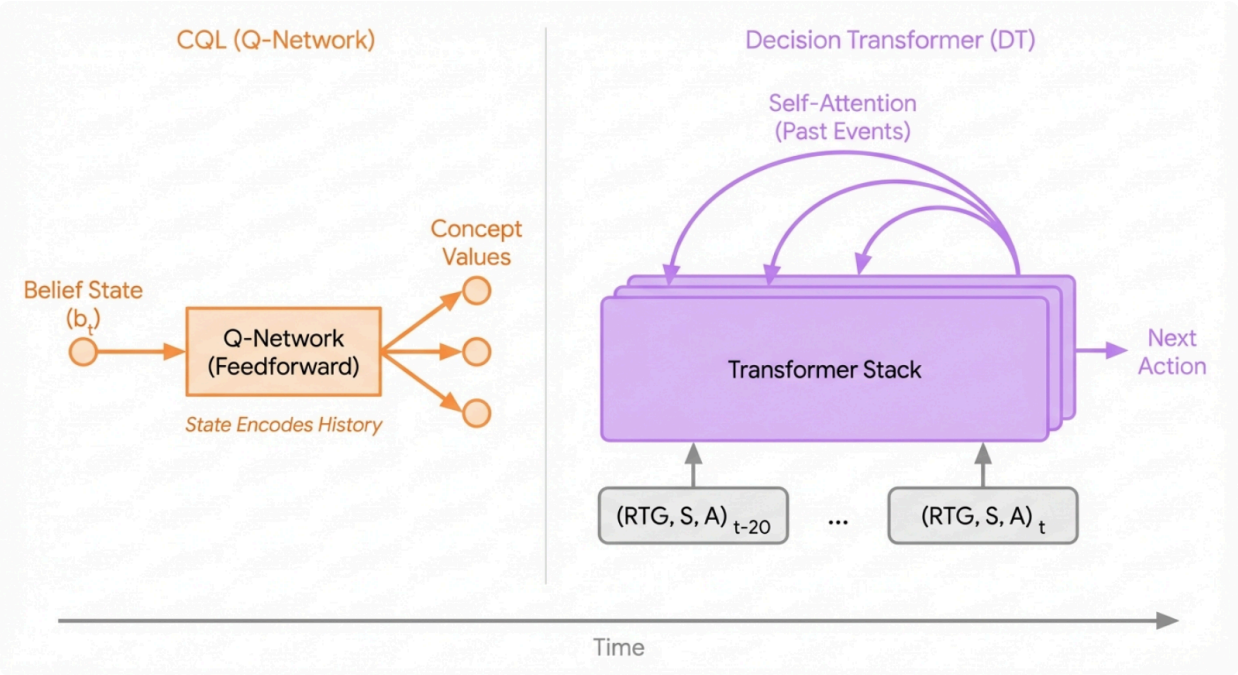
A key strength of DT in offline RL is its ability to "stitch" together optimal subsequences. If one student trajectory shows optimal performance on Concepts A and B but fails on C, and another trajectory fails on A but excels on C, the Transformer (conditioned on a high Return-to-Go) can learn to predict the optimal action sequence $A \to B \to C$ by attending to the high-reward segments of different histories.

Why DT dominates in Spacing Problems

Interleaving and spacing are fundamentally **non-Markovian** problems regarding the raw observations. The optimal time to review a concept depends on the *entire history* of interactions with that concept (e.g., "Was it reviewed yesterday? And 3 days before that? And 1 week before that?").

- **CQL** relies on the Belief State b_t to compress this entire history into a single vector. If the TD-BKT model is imperfect, CQL loses the history.
- **DT** uses **Self-Attention**. It can look back at a_{t-50} or a_{t-100} directly in the sequence context window. This allows it to learn complex temporal dependencies ("If the student struggled with Concept A 20 steps ago, and hasn't seen it since, schedule it now") that might be lost in the compressed belief state.[22]

Offline RL Architectures: Conservative Q-Learning vs. Decision Transformer



Architectural divergence in handling curriculum history. (Left) CQL utilizes a Q-Network taking the current Belief State (b_t) to output values for concepts, relying on the state vector to encode all history. (Right) The Decision Transformer ingests a full trajectory of (Return-to-Go, State, Action) tokens. Its Self-Attention mechanism (highlighted arcs) allows it to directly attend to past events (e.g., a practice session 20 steps ago), natively capturing long-term spacing dependencies.

5.3 Handling Exploration-Exploitation

In Offline RL, "Exploration" is handled differently than in Online RL (where we use ϵ-greedy to try random actions).

- **Offline Training Phase:** The "exploration" is limited to the diversity of the dataset. We cannot explore new actions. We rely on the algorithm (CQL/DT) to generalize and find better combinations of known actions.
- **Online Deployment Phase:** When we deploy this model, we do not want high-variance exploration (risking student confusion). However, we need some exploration to verify our estimated Q-values.
    - **Strategy:** We use **Boltzman Exploration** (Softmax) with a low temperature

parameter τ on the masked logits. This allows the agent to occasionally select the 2nd or 3rd best valid concept (which are likely still good) to gather data, without selecting potentially disastrous random actions.

---

6. Reward Engineering: Optimizing for Long-Term Retention

Standard RL rewards agents for "winning" the current step (binary correctness y_t). In education, maximizing y_t encourages **Blocking** (asking the same easy question repeatedly), because the student will answer correctly due to short-term working memory. This "cramming" produces high immediate scores but near-zero long-term retention.

To optimize for **Spaced Interleaving**, we must decouple the *Reward* from the *Observation*. We need a reward function that pays out based on the **Long-Term Memory Strength** of the student. Since we cannot wait 30 days to give the agent a reward, we use a proxy model: **Half-Life Regression (HLR)**.

6.1 The Reward Proxy: Half-Life Regression (HLR)

HLR [24] is a psychometric model that estimates the "half-life" h of a memory—the time it takes for the probability of recall to drop to 50%. The probability of recall p at time t (after time Δ since the last review) is modeled as: $$p = 2^{-\frac{\Delta}{h}}$$The half-life h itself is dynamic. It increases every time the student successfully recalls the item (spacing effect).

$$h_{new} = h_{prev} \cdot 2^{\Theta \cdot x}$$

Where x represents features of the interaction (e.g., current difficulty, result).

6.2 The Shaped Reward Function: Delta-Strength

We define the reward r_t passed to the RL agent as the **Gain in Total System Memory Strength**.

Let S_t^k be the "Memory Strength" (or stability) of concept k at time t. In HLR, $S \approx \log_2(h)$.

$$r_t = \sum_{k-1}^{K} (S_t^k - S_{t-1}^k)$$

This summation captures the full dynamics of the system:

1. **Active Gain:** If the student practices concept k and succeeds, h^k increases significantly (e.g., from 2 days to 5 days). This yields a large positive reward.
2. **Passive Decay:** For all other concepts $j \neq k$ that were *not* practiced, their stability might remain constant, but their *retention probability* p drops as Δ increases.
    - *Alternative Formulation:* We can define reward based on the sum of recall probabilities: $r_t = \sum_{k-1}^{K} (p_t^k - p_{t-1}^k)$. In this case, not reviewing a decaying concept j results in a negative term $(p_t^j < p_{t-1}^j)$, penalizing the agent.

Behavioral Outcome: This reward function forces the agent to play a "whack-a-mole" game. It must constantly monitor the decay rates of all K concepts. It learns that neglecting Concept A for too long results in a massive penalty (probability drop), forcing it to schedule a review (Spacing) exactly when the memory is about to fade, which is the theoretical optimum for spaced repetition.

---

7. Experimental Framework and Benchmarking

To validate this system without risking student outcomes, we utilize a high-fidelity **Student Simulator**.

7.1 The Baseline: 0.75 Mastery Threshold

We compare our CRL agent against the industry-standard heuristic used in systems like Khan Academy or ALEKS (in simplified forms).

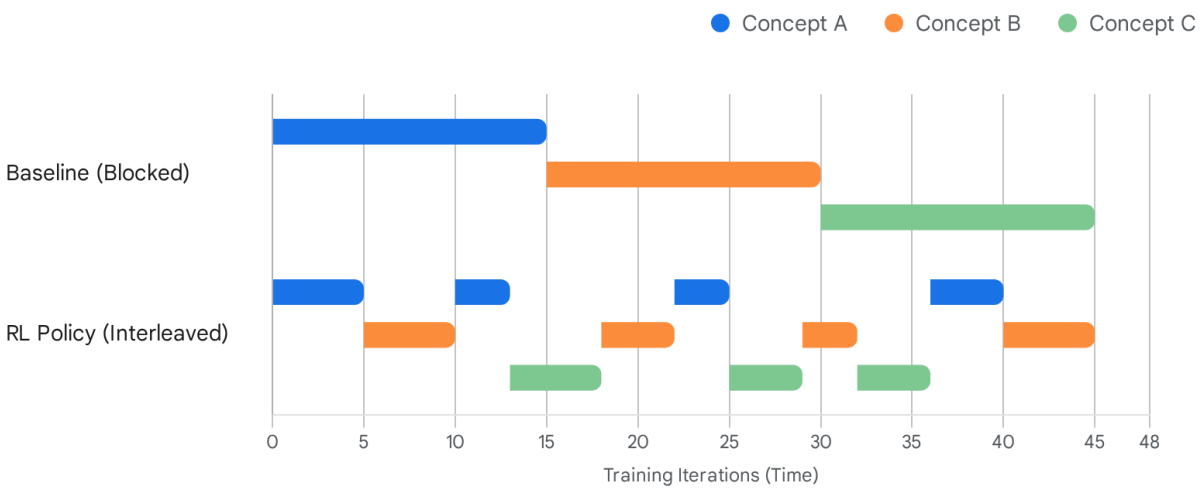
- **Algorithm:** The scheduler selects the current "active" concept. It generates problems for this concept until the student's moving average accuracy (or BKT mastery prob) exceeds **0.75**.
- **Transition:** Once threshold > 0.75, it marks the concept as "Mastered" and moves to the next concept in the linear prerequisite chain.
- **Critique:** This causes **Blocked Practice**. The student does AAAAA until mastery, then BBBBB. There is no spaced review of A.

7.2 Comparison Metrics

We run a simulation of 1000 students over a simulated "semester" (e.g., 1000 interaction

steps).

Metric	Definition	Hypothesis (RL vs Baseline)
Day-30 Retention	The average recall probability of all learned concepts 30 days after the course ends.	**RL >> Baseline.** The RL agent optimized for this specifically via spacing. Baseline ignores it.
Learning Efficiency	Number of steps required to reach >0.9 mastery on all concepts.	**RL > Baseline.** Interleaving reduces the "illusion of competence" and strengthens transfer, potentially requiring fewer total trials for robust learning.
Interleaving Score	A normalized entropy score of the concept sequence. High score = frequent switching. Low score = blocking.	**RL >> Baseline.** The Baseline score will be near 0 (perfect blocking). RL will naturally learn to switch.
Prerequisite Violations	% of actions where a student faces a concept without knowing parents.	**Both = 0%.** Due to the hard Action Masking constraint applied to both.



Comparison of instructional sequences. (Top) The Baseline '0.75 Threshold' scheduler creates Blocked Practice: Concept A is repeated until mastery, then B, then C. There is no return to A. (Bottom) The Curriculum RL (Decision Transformer) generates Spaced Interleaving: Concepts are mixed. Concept A is revisited after intervals of B and C, optimizing the spacing effect to combat memory decay.

Data sources: Int. Conf. Intelligent Tutoring Systems, UPenn Repository, Neurology

---

8. Implementation Guide

8.1 Data Pipeline Requirements

To implement this, one cannot simply use raw logs. The data processing pipeline is critical:

1. **Ingestion:** Load raw interaction logs (User, Item, Result, Timestamp).
2. **Graph Construction:** Define the adjacency matrix A_{prereq} for the Knowledge Graph.
3. **State Annotation (TD-BKT):** Run the TD-BKT algorithm forward through the data. For each timestep t, append the calculated Belief Vector b_t to the record.
4. **Reward Annotation (HLR):** Train an HLR model on the full dataset. Then, replay the data to calculate the "Counterfactual Memory Strength" at each step to generate the reward signal r_t.
5. **Tokenization (For DT):** Discretize the continuous rewards and states if necessary, or

map them to embeddings.

8.2 Model Training Specifications

- **Decision Transformer:**
  - **Context Length:** 20-50 steps (to capture relevant history for spacing).
  - **Embeddings:** Linear projection for continuous state s_t, learned embeddings for discrete action a_t.
  - **Training Objective:** Minimize Cross-Entropy loss for action prediction, conditioned on history and target return.
- **CQL:**
  - **Architecture:** Multi-Layer Perceptron (MLP) for Q-network.
  - **Alpha (Conservatism):** Tuning this hyperparameter is vital. High α leads to safer, "closer to teacher" policies. Low α allows more aggressive interleaving but risks instability.

9. Conclusion

Implementing a spaced interleaving system using Curriculum Reinforcement Learning represents a paradigm shift from **heuristic-based** to **optimization-based** education. By modeling the student as a POMDP with **TD-BKT** belief states, we rigorously capture the temporal dynamics of learning and forgetting that standard models miss. By employing **Offline RL** (specifically Decision Transformers), we safely extract optimal interleaving strategies from historical data, effectively "cloning" the successes of the best students while avoiding the failures of the worst. Finally, by masking actions with a **Knowledge Graph** and shaping rewards via **Half-Life Regression**, we ensure the system is both pedagogically valid and optimized for the true goal of education: robust, long-term mastery. This architecture offers a scalable, data-driven path to personalized learning that adapts not just to *what* a student knows, but *how* they forget.

---

10. References

- [4] TD-BKT: Temporal Difference Bayesian Knowledge Tracing.
- [17] Conservative Q-Learning (CQL) for Offline RL.
- [19] Decision Transformers for Sequence Modeling in RL.
- [9] Action Masking and Constrained RL.

- [24]
  Half-Life Regression and Spaced Repetition Optimization.
- [26]
  Mastery Learning Thresholds and Baselines.

Works cited

1. Experiment 1 -The average distance of belief b to the true state S for... - ResearchGate, accessed January 25, 2026, https://www.researchgate.net/figure/Experiment-1-The-average-distance-of-belief-b-to-the-true-state-S-for-each-of-the-four_fig1_353835718
2. BKT-POMDP: Fast Action Selection for User Skill Modelling over Tasks with Multiple Skills, accessed January 25, 2026, https://www.researchgate.net/publication/353835718_BKT-POMDP_Fast_Action_Selection_for_User_Skill_Modelling_over_Tasks_with_Multiple_Skills
3. Reinforcement Learning of POMDPs using Spectral Methods, accessed January 25, 2026, http://proceedings.mlr.press/v49/azizzadenesheli16a.pdf
4. Integrating Temporal Information Into Knowledge Tracing: A Temporal Difference Approach - IEEE Xplore, accessed January 25, 2026, https://ieeexplore.ieee.org/document/8357440/
5. Integrating Temporal Information Into Knowledge Tracing - IEEE Xplore, accessed January 25, 2026, https://ieeexplore.ieee.org/iel7/6287639/8274985/08357440.pdf
6. Time-dependant Bayesian knowledge tracing—Robots that model user skills over time - NIH, accessed January 25, 2026, https://pmc.ncbi.nlm.nih.gov/articles/PMC10925631/
7. Time-dependant Bayesian knowledge tracing—Robots that model user skills over time - Frontiers, accessed January 25, 2026, https://www.frontiersin.org/journals/robotics-and-ai/articles/10.3389/frobt.2023.1249241/full
8. Enhancing Robot Assistive Behaviour with Reinforcement Learning and Theory of Mind, accessed January 25, 2026, https://arxiv.org/html/2411.07003v1
9. Masking in Deep Reinforcement Learning - Boring Guy, accessed January 25, 2026, https://boring-guy.sh/posts/masking-rl/
10. A Framework for Budget-Constrained Zero-Day Cyber Threat Mitigation: A Knowledge-Guided Reinforcement Learning Approach - NIH, accessed January 25, 2026, https://pmc.ncbi.nlm.nih.gov/articles/PMC12787384/
11. Computers and Chemical Engineering, accessed January 25, 2026, https://d-nb.info/128069369X/34
12. GNN-DT: Graph Neural Network Enhanced Decision Transformer for Efficient Optimization in Dynamic Environments - arXiv, accessed January 25, 2026, https://arxiv.org/pdf/2502.01778
13. Depiction of the Decision Transformer and its inputs, adapted from [21]. - ResearchGate, accessed January 25, 2026, https://www.researchgate.net/figure/Depiction-of-the-Decision-Transformer-and-its-inputs-adapted-from-21_fig1_389533897

14. Offline Reinforcement Learning for Learning to Dispatch for Job Shop Scheduling - arXiv, accessed January 25, 2026, https://arxiv.org/html/2409.10589v1
15. A Study of Generalization in Offline Reinforcement Learning - OpenReview, accessed January 25, 2026, https://openreview.net/pdf?id=0Pbh8u7oXi
16. A Survey on Offline Reinforcement Learning: Taxonomy, Review, and Open Problems - IEEE Xplore, accessed January 25, 2026, https://ieeexplore.ieee.org/iel7/5962385/10623582/10078377.pdf
17. Conservative Q-Learning for Offline Reinforcement Learning - NeurIPS, accessed January 25, 2026, https://papers.neurips.cc/paper_files/paper/2020/file/0d2b2061826a5df3221116a5085a6052-Paper.pdf
18. Enhancing Offline Reinforcement Learning with Curriculum Learning-Based Trajectory Valuation - arXiv, accessed January 25, 2026, https://arxiv.org/html/2502.00601v1
19. Analysis and optimization of student learning paths based on CRNN and sequential data, accessed January 25, 2026, https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0331491
20. Cross-Episodic Curriculum for Transformer Agents - OpenReview, accessed January 25, 2026, https://openreview.net/pdf?id=afKnrwJBAI
21. Decision Transformer: Reinforcement Learning via Sequence Modeling - OpenReview, accessed January 25, 2026, https://openreview.net/pdf?id=a7APmM4B9d
22. StARformer: Transformer with State-Action-Reward Representations for Visual Reinforcement Learning, accessed January 25, 2026, https://www.ecva.net/papers/eccv_2022/papers_ECCV/papers/136990455.pdf
23. Sample-efficient Imitative Multi-token Decision Transformer for Real-world Driving - arXiv, accessed January 25, 2026, https://arxiv.org/html/2407.02508v2
24. A Trainable Spaced Repetition Model for Language Learning - Duolingo Research, accessed January 25, 2026, https://research.duolingo.com/papers/settles.acl16.pdf
25. DRL-SRS: A Deep Reinforcement Learning Approach for Optimizing Spaced Repetition Scheduling - MDPI, accessed January 25, 2026, https://www.mdpi.com/2076-3417/14/13/5591
26. DATA QUANTITY AND DATA CHARACTERISTICS FOR MODELING APPROACHES IN EDUCATIONAL DATA MINING Stefan Slater A DISSERTATION in Educa - University of Pennsylvania, accessed January 25, 2026, https://repository.upenn.edu/bitstreams/d811324f-2845-4d74-bbba-7364f34edc60/download
27. Junior Neurology Residents Achieve Competency but Not Mastery After a Brief Acute Ischemic Stroke, accessed January 25, 2026, https://www.neurology.org/doi/pdfdirect/10.1212/NE9.0000000000200071