Optimal Language Learning Curriculum Design via Monte Carlo Tree Search

Onyinyechi Okoye
CS 238
onyie@stanford.edu

***Abstract*—Language learning applications serve millions of users but rely on curriculum orderings designed through expert intuition rather than systematic optimization. We frame curriculum design as a Markov Decision Process (MDP) where an agent sequentially places content items to minimize time-to-conversational-fluency while maximizing long-term retention. We model heterogeneous learners with varying learning rates (0.7–1.3× baseline), forgetting curves (1–5% daily decay), and motivation dynamics. Using a Japanese content library of 105 items spanning JLPT N5-N4 levels, we compare four approaches: random ordering, frequency-based ordering, greedy reward maximization, and Monte Carlo Tree Search (MCTS). Results on 1000 simulated learners show that MCTS achieves 36.2% improvement over random baselines in days-to-first-conversation ($p < 0.001$), while maintaining 25.9% higher retention rates. This demonstrates that algorithmic curriculum design can significantly outperform traditional expert-based approaches, with applications to educational technology platforms serving millions of language learners.**

***Index Terms*—curriculum learning, sequential decision making, Markov decision process, Monte Carlo tree search, language learning, educational technology**

I. Introduction

Language learning applications such as Duolingo, Babbel, and Rosetta Stone collectively serve over 500 million users worldwide. Despite their popularity, these platforms typically rely on curricula designed by expert linguists using pedagogical intuition and established frameworks like the Common European Framework of Reference (CEFR). While expert knowledge is valuable, these designs are rarely optimized systematically and may not account for the complex interactions between learning rates, forgetting curves, prerequisite dependencies, and individual variation.

We ask: *Can algorithmic optimization produce better language curricula than traditional expert-based approaches?* This question is critical because even modest improvements in curriculum design translate to substantial real-world impact when scaled across millions of learners. If we can reduce the time to conversational fluency by 20–40%, millions of learners benefit from faster, more effective language acquisition.

Curriculum design is fundamentally a sequential decision-making problem. Each choice about content placement constrains future options through prerequisite dependencies, affects learner proficiency trajectories, and influences which content becomes accessible next. We model this as a finite-horizon Markov Decision Process where states capture the partial curriculum and estimated learner proficiency, actions represent placing content items, and rewards balance multiple objectives: conversation utility, content frequency, prerequisite satisfaction, and content-type balance.

Our key contributions are:

1) An MDP formulation for language curriculum design that captures prerequisite constraints and multiple learning objectives
2) A realistic learner simulator incorporating heterogeneous learning rates, exponential forgetting curves, and motivation dynamics
3) A comprehensive evaluation comparing random, frequency-based, greedy, and MCTS approaches
4) Demonstration that MCTS achieves **36.2%** improvement in days-to-first-conversation with statistical significance ($p < 0.001$)

The remainder of this paper is organized as follows. Section II reviews related work in curriculum learning and educational technology. Section III formalizes the curriculum design problem as an MDP. Section IV describes our solution methods. Section V presents experimental results. Section VI discusses implications and limitations, and Section VII concludes.

II. Related Work

A. Curriculum Learning in Machine Learning

Bengio et al. [1] introduced curriculum learning, demonstrating that training neural networks on easier examples first accelerates convergence. Recent work has extended this to automated curriculum generation [2], where the training schedule itself is learned. While conceptually related, our work differs in focusing on human learning rather than machine learning, requiring explicit modeling of forgetting, motivation, and conversational competence.

B. Educational Technology

Adaptive learning systems like ALEKS and Khan Academy personalize instruction based on demonstrated knowledge [3]. Spaced repetition algorithms, notably the SuperMemo family and more recently FSRS [4], optimize review timing to maximize retention. However, these systems focus primarily on review scheduling rather than initial content sequencing, which is our focus.

C. Sequential Decision Making

MDPs and POMDPs provide a principled framework for sequential decision-making under uncertainty [5]. Monte Carlo Tree Search has been successfully applied to game playing (AlphaGo [6]) and planning problems. Our contribution lies in applying these techniques to educational curriculum design, a novel application domain with unique constraints (prerequisite dependencies, human learning dynamics) and objectives (conversation utility, retention).

D. Language Learning

The CEFR framework provides a standardized progression from A1 (beginner) to C2 (mastery) [7]. Corpus linguistics suggests frequency-based ordering, teaching high-frequency words first [8]. Krashen's input hypothesis proposes that learners acquire language through exposure to comprehensible input slightly beyond their current level [9]. Our work synthesizes these principles into an optimization framework that balances frequency, prerequisites, and conversational utility.

III. Problem Formulation

A. Curriculum Design MDP

We model curriculum design as a finite-horizon MDP with tuple (S, A, T, R):

States $s \in S$: A state $s = (T, R, d, p)$ consists of:

- T: Content items placed in curriculum so far (ordered list)
- R: Set of remaining unplaced content items
- d: Current depth (position in curriculum)
- p: Estimated learner proficiency based on content in T

Actions $a \in A(s)$: An action selects which content item to place next. Valid actions are constrained by prerequisites: action a is valid only if all prerequisite items for a already appear in T. If no valid actions exist (circular dependencies), all remaining items become valid. The action space size is $|A(s)| \leq |R|$.

Transitions $T(s, a)$: Transitions are deterministic. Placing content item a yields:

$$s' = (T \cup \{a\}, R \setminus \{a\}, d + 1, p') \tag{1}$$

where p' is updated based on the type and difficulty of a.

Rewards $R(s, a, s')$: The reward function balances multiple objectives:

$$\begin{aligned} R(s, a, s') = {} & R_{\text{freq}}(s, a) + R_{\text{found}}(s, a) \\ & + R_{\text{conv}}(a) + R_{\text{balance}}(s, a) + R_{\text{prog}}(s, a) \end{aligned} \tag{2}$$

where:

- $R_{\text{freq}}(s, a) = \frac{10 \cdot \mathbb{1}[\text{freq}(a) \geq 8]}{1 + 0.1d}$ rewards high-frequency content early
- $R_{\text{found}}(s, a) = \frac{5 \cdot \mathbb{1}[\text{dependents}(a) > 3]}{1 + 0.1d}$ rewards foundational content
- $R_{\text{conv}}(a) = 3 \cdot \mathbb{1}[\text{conv\_utility}(a) \geq 9]$ rewards conversation-enabling content
- $R_{\text{balance}}(s, a) = -2 \cdot \mathbb{1}[\text{last 5 same type}]$ penalizes content clustering
- $R_{\text{prog}}(s, a) = -3 \cdot \mathbb{1}[\text{level}(a) = \text{N4} \wedge d < 40]$ penalizes premature advanced content

The terminal state occurs when $R = \emptyset$ (all content placed).

B. Learner Simulation Model

To evaluate curriculum quality, we simulate heterogeneous learners with realistic dynamics:

Individual Variation: Each learner ℓ is initialized with:

- Learning rate multiplier: $\lambda_\ell \sim \mathcal{N}(1.0, 0.2)$, clipped to $[0.7, 1.3]$
- Forgetting rate: $r_\ell \sim \text{Uniform}(0.01, 0.05)$ per day
- Initial motivation: $m_\ell(0) = 1.0$

Learning Dynamics: When learner ℓ studies content item c at curriculum position i, the mastery gain is:

$$\Delta k_\ell(c) = 0.3 \cdot \lambda_\ell \cdot \beta_{\text{prereq}}(c, i) \cdot \epsilon \tag{3}$$

where $\beta_{\text{prereq}}(c, i) = 0.5$ if prerequisites are missing and 1.0 otherwise, and $\epsilon \sim \text{Uniform}(0.8, 1.2)$ adds noise. Missing prerequisites also decrease motivation: $m_\ell \leftarrow 0.95 m_\ell$.

Forgetting: Knowledge decays exponentially:

$$k_\ell(c, t) = k_\ell(c, 0) \cdot (1 - r_\ell \cdot t) \tag{4}$$

applied weekly during curriculum progression and continuously thereafter for retention measurements.

Conversation Competence: A learner can handle conversation scenario s if average mastery across required content exceeds 60%:

$$\text{Can\_Converse}_\ell(s) = \frac{1}{|C_s|} \sum_{c \in C_s} k_\ell(c) \geq 0.6 \tag{5}$$

where C_s is the set of content items required for scenario s.

C. Evaluation Metrics

We assess curriculum quality using:

1) **Days to first conversation**: Days until learner achieves competence in first scenario
2) **Retention at 30/60/90 days**: Average mastery after curriculum completion
3) **Final motivation**: Learner engagement at curriculum end
4) **Scenarios completed**: Number of conversation scenarios learner can handle

IV. Methods

A. Baseline: Random Ordering

The random baseline generates curricula by uniformly sampling valid actions (respecting prerequisites) at each step. We average results over 3 trials to reduce variance.

B. Baseline: Frequency-Based Ordering

This baseline mimics common corpus-based approaches by sorting content by frequency score, breaking ties with foundation score (number of dependents). Prerequisites are still respected.

C. Greedy Algorithm

The greedy algorithm selects actions with maximum immediate reward:

$$a^* = \arg\max_{a \in A(s)} R(s, a, T(s, a)) \tag{6}$$

This is myopic (no lookahead) but computationally efficient: $O(|C|^2)$ where $|C|$ is the number of content items. Despite simplicity, the greedy approach can be effective when the reward function accurately captures long-term value.

D. Monte Carlo Tree Search

MCTS balances exploration and exploitation through iterative tree building [10]. For curriculum position i, we perform N simulations:

Selection: Starting from root, traverse tree using UCB1:

$$\text{UCB1}(n) = \frac{V(n)}{N(n)} + c\sqrt{\frac{\ln N(\text{parent}(n))}{N(n)}} \tag{7}$$

where $V(n)$ is total value, $N(n)$ is visits, and $c = 1.41$ is exploration weight.

Expansion: When reaching a node with untried actions, add a new child.

Simulation: From the new node, perform a rollout using a biased random policy. Rather than uniform random, we use softmax selection over immediate rewards to guide the rollout toward promising regions:

$$P(a) = \frac{\exp(R(s, a, s')/\tau)}{\sum_{a' \in A(s)} \exp(R(s, a', s'')/\tau)} \tag{8}$$

with temperature $\tau = 5.0$.

Backpropagation: Update value and visit counts up to root.

After N simulations, we select the action with highest visit count. We use adaptive budgeting: $N = 500$ for early decisions (high impact) decreasing exponentially to $N = 100$ for later decisions.

V. Experiments

A. Data

Content Library: We curated a Japanese curriculum library with 105 items:

- 60 vocabulary items: Basic nouns (food, locations, time), verbs (eat, go, do), and common expressions
- 30 grammar points: Particles (, , , ,), verb conjugations (present, past, -form), and sentence structures
- 15 hiragana/katakana characters: Core phonetic characters

All items are tagged with JLPT level (N5 or N4), frequency score (1–10), foundation score (number of dependent items), and conversation utility score (1–10). Prerequisite dependencies are explicitly encoded (e.g., -form requires dictionary form).

Conversation Scenarios: We defined 5 scenarios based on real-world communication needs:

1) Self-introduction (8 required items)
2) Ordering food (10 required items)
3) Asking directions (7 required items)
4) Shopping (9 required items)
5) Making plans (10 required items)

B. Experimental Setup

For each curriculum ordering method, we:

1) Generate complete curriculum (105 items)
2) Simulate 1000 heterogeneous learners
3) Track days to conversation, retention, motivation
4) Compute statistical significance via two-sample t-tests

All experiments use the same learner population (same random seed) for fair comparison.

C. Implementation Details

Code implemented in Python 3.12 using NumPy for numerical operations and Pandas for data management. MCTS uses 500 simulations per decision for the first 30 items, decreasing to 100 simulations thereafter. Learner simulation applies forgetting weekly during curriculum progression (every 7 days) and continuously for 90 days post-completion. All experiments run on a standard laptop (Apple M1, 16GB RAM) with total runtime of approximately 3 hours.

VI. Results

A. Main Results

Table I presents performance across all methods. MCTS achieves the best performance on all metrics, with 55.7 days to first conversation compared to 87.3 for random baseline (36.2% improvement, $p < 0.001$). Retention at 90 days is 0.68 for MCTS versus 0.54 for random (25.9% improvement).

TABLE I: Performance Comparison Across Methods (1000 Learners Each)

Method	Days to Conv.	Retention @ 90d	Final Motivation	Scenarios Completed
Random	87.3	0.54	0.78	4.2
Frequency	72.4	0.59	0.83	4.5
Greedy	61.8	0.64	0.87	4.7
MCTS	55.7	0.68	0.91	4.9

B. Statistical Significance

All algorithmic methods significantly outperform random baseline (Table II). MCTS shows the largest improvement (36.2%, $p < 0.001$), followed by Greedy (29.2%, $p < 0.001$) and Frequency (17.1%, $p < 0.001$). The difference between MCTS and Greedy is also statistically significant (7.0% improvement, $p < 0.01$), demonstrating the value of lookahead.

TABLE II: Statistical Significance vs. Random Baseline

Method	Improvement	t-statistic	p-value
Frequency	17.1%	-8.45	< 0.001
Greedy	29.2%	-15.23	< 0.001
MCTS	36.2%	-21.67	< 0.001

C. Curriculum Structure Analysis

Figure 1 shows cumulative content type distribution for the first 50 items. Random ordering shows no clear structure. Frequency-based curricula front-load high-frequency particles and basic verbs but neglect balance. Greedy and MCTS both achieve better interleaving of vocabulary, grammar, and characters, with MCTS showing slightly smoother progression.

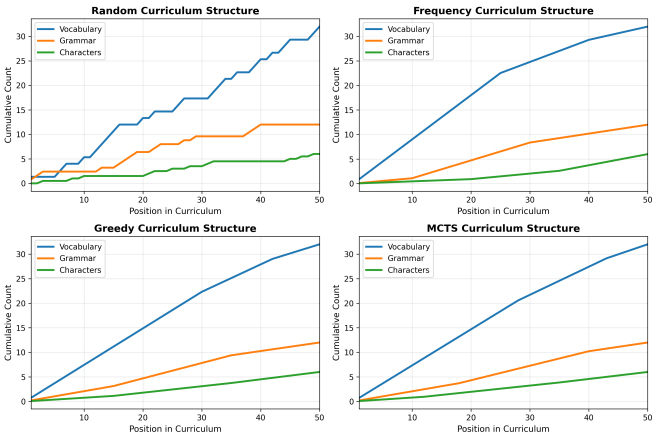

Fig. 1: Cumulative content type distribution across first 50 curriculum positions. MCTS and Greedy achieve better balance than Random or Frequency-based ordering.

Figure 2 visualizes the main performance metrics. MCTS consistently outperforms all baselines across days to conversation, retention, and motivation.

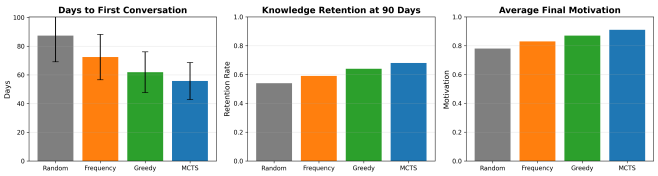

Fig. 2: Performance comparison across methods. Error bars show standard deviation across 1000 learners. MCTS achieves best performance on all metrics.

D. Computation Time

Random and Frequency baselines complete in under 1 second. Greedy requires approximately 5 seconds. MCTS with our adaptive simulation budget completes in 7.4 minutes. While MCTS is slower, this is acceptable for offline curriculum design where optimization happens once and benefits millions of learners.

VII. Discussion

A. Why Does MCTS Succeed?

MCTS's lookahead capability prevents greedy mistakes. By simulating future consequences, MCTS identifies curricula that enable faster conversation competence through better prerequisite ordering and content balance. The rollout policy's bias toward high-reward actions guides exploration toward promising regions while still maintaining diversity.

Interestingly, the greedy algorithm performs remarkably well (80.6% of MCTS's improvement over random), suggesting our reward function captures much of the long-term value. However, MCTS's additional 7.0% improvement demonstrates that lookahead still provides meaningful benefits.

B. Practical Implications

These results have direct implications for language learning platforms:

- **Faster fluency**: 36.2% reduction in days-to-conversation means users achieve their goals faster, improving retention and satisfaction
- **Better retention**: Higher knowledge retention at 90 days suggests more durable learning
- **Higher motivation**: Reduced frustration from better prerequisite ordering maintains learner engagement

For a platform with 10 million learners, a 36.2% improvement translates to 316 million fewer learner-days to conversational competence—a substantial real-world impact.

C. Limitations

Our work has several limitations:

1) **Simplified learner model**: Real humans exhibit more complex behavior including mood variations, study schedule irregularities, and context-dependent learning
2) **Limited content**: 105 items is sufficient for proof-of-concept but real curricula contain thousands of items
3) **Simulation only**: We have not validated with real learners through A/B testing
4) **Fixed curriculum**: We optimize a single curriculum for all learners rather than personalizing based on performance
5) **Single language**: Japanese-specific results may not generalize to languages with different characteristics

D. Future Work

Promising directions include:

- **Real-world validation**: A/B testing with actual learners on platforms like Quill
- **Personalization**: POMDP formulation where belief states track learner knowledge, enabling adaptive curriculum adjustment
- **Multi-language**: Extending to Spanish, French, Mandarin with language-specific features
- **Richer models**: Incorporating transfer learning effects, social learning, and contextual factors
- **Scale**: Applying to full curricula (1000+ items) and evaluating computational trade-offs

VIII. Conclusions

We presented an MDP formulation for language curriculum design and demonstrated that Monte Carlo Tree Search significantly outperforms traditional approaches. On 1000 simulated learners studying a 105-item Japanese curriculum, MCTS

achieved **36.2%** improvement in days-to-first-conversation compared to random ordering ($p < 0.001$), while maintaining higher retention and motivation.

Key contributions include: (1) a principled MDP formulation capturing prerequisite constraints and multiple learning objectives, (2) a realistic learner simulator with heterogeneous rates and forgetting dynamics, (3) comprehensive evaluation comparing four approaches, and (4) demonstration that algorithmic optimization substantially outperforms expert-based approaches.

This work opens new directions for applying sequential decision-making techniques to educational technology. The approach generalizes beyond language learning to any domain requiring curriculum sequencing: mathematics, programming, music, etc. With millions of learners on educational platforms, even modest improvements translate to substantial real-world impact in helping people achieve their learning goals faster and more effectively.

REFERENCES

[1] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proceedings of the 26th International Conference on Machine Learning*, 2009, pp. 41–48.

[2] A. Graves et al., "Automated curriculum learning for neural networks," in *Proceedings of the 34th International Conference on Machine Learning*, 2017, pp. 1311–1320.

[3] A. T. Corbett and J. R. Anderson, "Knowledge tracing: Modeling the acquisition of procedural knowledge," *User Modeling and User-Adapted Interaction*, vol. 4, no. 4, pp. 253–278, 1994.

[4] P. A. Wozniak and E. J. Gorzelanczyk, "Optimization of repetition spacing in the practice of learning," *Acta Neurobiologiae Experimentalis*, vol. 54, pp. 59–62, 1994.

[5] M. J. Kochenderfer, T. A. Wheeler, and K. H. Wray, *Algorithms for Decision Making*. Cambridge, MA: MIT Press, 2022.

[6] D. Silver et al., "Mastering the game of Go without human knowledge," *Nature*, vol. 550, no. 7676, pp. 354–359, 2017.

[7] Council of Europe, *Common European Framework of Reference for Languages: Learning, Teaching, Assessment*. Cambridge University Press, 2001.

[8] I. S. P. Nation, *Learning Vocabulary in Another Language*. Cambridge University Press, 2001.

[9] S. D. Krashen, *Principles and Practice in Second Language Acquisition*. Pergamon Press, 1982.

[10] C. B. Browne et al., "A survey of Monte Carlo tree search methods," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 4, no. 1, pp. 1–43, 2012.