A Queueing Network Model for Spaced Repetition
Siddharth Reddy, Igor Labutov, Siddhartha Banerjee
{sgr45, ili4, shbanerjee}@cornell.edu

Introduction

The ability to retain a large number of new ideas in memory is an essential component of human learning. In recent times, there has been a growing body of work that attempts to engineer this process – creating tools that enhance the learning process by building on the scientific understanding of human memory. Flashcards are one such tool that use the idea of spaced repetition to overcome the human forgetting curve. Though they have been around for a while in the physical form, a new generation of spaced repetition software such as SuperMemo, Anki, and Mnemosyne allow a much greater degree of control and monitoring of the process. As these software applications grow popular, there is a need for formal models for reasoning about and optimizing their operations. In this work, we use ideas from queueing theory to develop such a formal model for one of the simplest and oldest spaced repetition systems: the Leitner system.

Key Contributions

- Formalize the Leitner spaced repetition system using ideas from queueing theory
- Validate the exponential forgetting curve on large-scale log data from the Mnemosyne flashcard software
- Verify key predictions of our queueing model using a controlled experiment on Amazon Mechanical Turk

Memory Model

We use a variant on the exponential forgetting curve to model the decay of human memory over time. The probability of a student recalling an item is as follows.

\[ P[\text{recall}] = \exp \left( -\theta \frac{D}{s} \right) \]

where \( \theta \) is the item difficulty, \( D \) is time elapsed since previous review, and \( s \) is memory strength.

Leitner Queue Network

New items arrive into deck 1 according to a Poisson process with rate \( \lambda_{ext} \). The routing probability matrix is \( P \), where

\[
P_{ij} = \begin{cases} 
   P[\text{recall} \mid s = i, j] & \text{if } i < n \land j = i + 1 \\
   1 - P[\text{recall} \mid s = i, j] & \text{if } (i > 1 \land j = i - 1) \lor i = j = 1 \\
   0 & \text{otherwise}
\end{cases}
\]

Items exit the system from deck \( n \) with probability \( P[\text{recall} \mid s = n, j] \). The service rate for deck \( i \) is indicated by \( \mu_i \), and the user's work rate budget (e.g., the maximum number of items the user can review per day) is given by \( U \). We are interested in finding \( \lambda_{ext} \) that maximize the steady-state throughput of the system such that the budget constraint \( \sum \mu_i \leq U \) is satisfied. Formally, we must solve the following static planning problem.

\[
\begin{align*}
\text{maximize} & \quad \lambda_{ext} \\
\text{subject to} & \quad \sum \mu_i \leq U \\
& \quad \mu_i \geq 0 \quad \forall i \\
& \quad \lambda_{ext} < \mu_i \quad \forall i
\end{align*}
\]

The algorithm for selecting the next item to present to the user is simple: sample deck \( i \) with probability \( \frac{\lambda_{ext}}{\sum \mu_i} \), and select the item at the top of the sampled deck.

Experiments

Full paper and code available at http://siddharth.io/leitnerq