# Adaptive Quiz Generation Using Thompson Sampling

Fuhua (Oscar) Lin, PhD

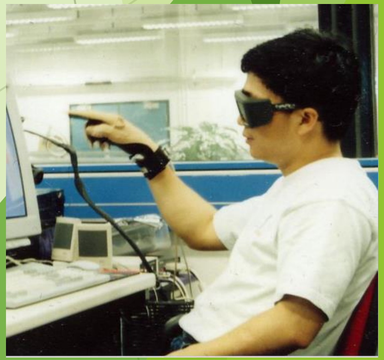
Athabasca University, Canada

Third Workshop eliciting Adaptive Sequences for Learning

(WASL 2020)

Cyberspace, 6 July, 2020

Co-located with the AIED 2020



# Outline

- Introduction
- Literature review
- The proposed method
  - Quiz Model and Student Model
  - Modeling the Quiz Generation Process
  - The Proposed Algorithm
- Implementation Plan
- Conclusions and Future Work

# Formative Assessment

- To make education more effective through identifying and closing the learning gaps.

# Principles of Formative Assessment
(Group, 1999).

- Integral part of instruction --- used **in real time** for guiding learning process.
- Student involvement.
  - for self-guidance and
  - to monitor their progress towards learning objectives.
- Constructive feedback to close the learning gaps.

# Formative Assessment in Online Learning

- Classroom
    - Face to face tutoring
    - Discussions
- Online learning environments
    - Learning analytics (LA) /Educational data mining (EDM)
    - Adaptive assessment --- Computerized assessment

# Adaptive Assessment

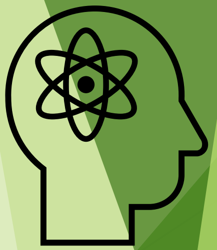
- Optimize the computerized assessment process so that students can receive an accurate evaluation in as little time as possible (Vie et al, 2012).

# Traditional Adaptive Assessment

- Based on
  - Item Response Theory (IRT) (Lord, 1980; Huang, et al. 2009)
  - Elo rating (Elo, 1978)
- Limitations
  - Complexity in implementation
  - The premise that different questions measure one common trait (Wainer, 2001).

# Our Method

- To design an algorithm that can accurately and quickly identify the lacking areas of knowledge of the student.
  - model the quiz sequence generation process as a Beta Bernoulli Bandit model and
  - solve it with Thompson Sampling algorithm which is one of multi-armed bandit algorithms and can use prior knowledge.

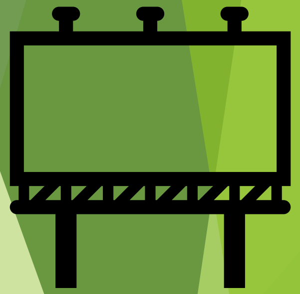


# Multi-Armed Bandit Algorithms

- Is named after a problem for a gambler who must decide which arm of a K-slot machine to pull to maximize his total reward in a series of trials.
- Are capable of negotiating exploration-exploitation trade-offs.
- Applied in real-world applications solving optimization problems
- Emerging applications of MAB algorithms for optimal learning material selection.

# Upper-Confidence Bound Algorithm

- Melesko and Novickij (2019) proposed and tested an alternative adaptive testing method based on Upper-Confidence Bound
  - Simple
  - Offering sub-linear regret.
  - Not random
  - Smart exploration
- Drawback
  - Cannot use prior knowledge

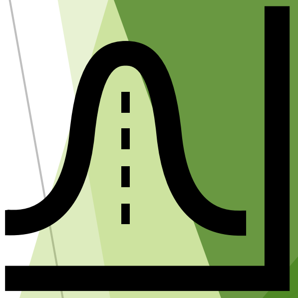


# Modelling

- The gambler <-> the system.
- A learning objective <-> Arm.
- Reward <-> answer by the student {0, 1}.
- Ending: reaching the maximum number of a quiz.
- Goal: explore the different topics and engage in focused questioning, exploiting those which are possibly in most need of further learning or remediation.

# Thompson Sampling Algorithm

- 1933 by William R. Thompson
- Effective in simulation and real-world applications
  - Smart exploration
- Main idea:
  - Bayesian approach to estimate the reward
  - To randomly select an arm according to the probability that it is optimal.



Thompson, William R. "On the likelihood that one unknown probability exceeds another in view of the evidence of two samples". *Biometrika*, 25(3–4):285–294, 1933.

# Thompson Sampling Algorithm

- A quiz sequence generation process
  - modeled as a Beta Bernoulli Bandit problem.
  - solved with Thompson Sampling algorithm as Thompson sampling can use prior knowledge of the student.

# Basic Models

- Domain Model: $\Delta = \{\delta_1, \delta_2, \dots, \delta_n\}$, $\delta_i$ is called knowledge unit (KU).
- Assessment Model $\mathbb{A}$
  - $LO(i) = \{lo(i,1), lo(i,2), \dots, lo(i,j), \dots lo(i,n_i)\}$. ($i$ = 1, 2, …, $K$). $lo(i,j)$ is $j^{th}$ learning objective in $\delta_i$.
  - For $lo(i,j)$, we design a set of assessment questions.
- Quiz Model
  - $Quiz = \{q(1), q(2), \dots, q(i), \dots q(m)\}$,
  - Each question is tagged with a set of tags including corresponding KUs, learning objectives, and feedback.

# *Learner model*

- Be represented as a time-series matrix where
  - rows --- learning objectives,
  - columns --- discrete times,
  - the value --- the probability that the student can answer the questions of the learning objective correctly.
- Record all the answers.

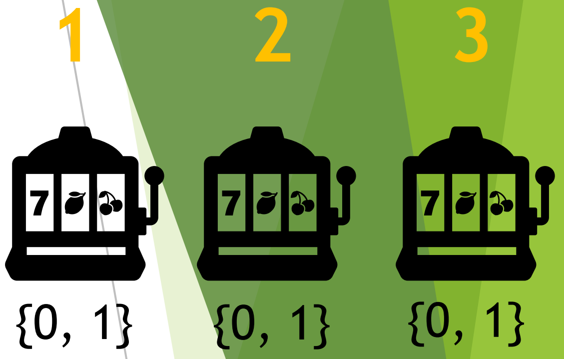
$$\begin{array}{rc} & \begin{array}{ccccc} t_0 & t_1 & t_2 & \ldots\ldots & t_m \end{array} \\ \Lambda_1 = \begin{array}{c} \delta_1 \\ \delta_2 \\ \delta_3 \\ . \\ \delta_n \end{array} & \left(\begin{array}{ccccc} 0.0 & 0.2 & 0.5 & & 1.0 \\ 0.0 & 0.0 & 0.3 & \cdots & 1.0 \\ 0.0 & \ldots. & 0.0 & & 1.0 \\ & \vdots & & \ddots & \vdots \\ 0.0 & 0.0 & 0.0 & \cdots & 1.0 \end{array}\right) \end{array}$$

$$\begin{array}{rc} & \begin{array}{ccccc} t_0 & t_1 & t_2 & \ldots\ldots & t_m \end{array} \\ LO_1 = \begin{array}{c} lo_{11} \\ lo_{12} \\ lo_{13} \\ . \\ lo_{1n} \end{array} & \left(\begin{array}{ccccc} 0.0 & 0.5 & 0.8 & & 1.0 \\ 0.0 & 0.4 & 0.6 & \cdots & 1.0 \\ 0.0 & \ldots. & 0.5 & & 1.0 \\ & \vdots & & \ddots & \vdots \\ 0.0 & 0.3 & 0.7 & \cdots & 1.0 \end{array}\right) \end{array}$$

$$\begin{array}{rc} & \begin{array}{ccccc} t_0 & t_1 & t_2 & \ldots\ldots & t_m \end{array} \\ A_{11} = \begin{array}{c} a_{111} \\ a_{112} \\ a_{113} \\ . \\ a_{11n} \end{array} & \left(\begin{array}{ccccc} 0 & 1 & 1 & & 1 \\ 0 & 1 & 0 & \cdots & 1 \\ 0 & .. & 0 & & 1 \\ & \vdots & & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{array}\right) \end{array}$$

# Bernoulli Bandit problem

- *K* actions: $\{1, \ldots, K\}$
- Rewards: {0, 1}
  - when played, an action $k \in \{1, \ldots, K\}$ produces a reward $r_t$ *of*
    - 1 with success probability $\theta_k \in [0, 1]$
    - 0 with probability 1-$\theta_k \in [0, 1]$.
  - $\theta_k$ **success probability** or **mean reward**.
- $(\theta_1, \ldots, \theta_K)$
  - unknown to the agent, fixed over time
  - can be learned by experimentation, denoted their **estimated** values as: $(\hat{\theta}_1, \hat{\theta}_2, \ldots, \hat{\theta}_K)$
- The objective is to **maximize** $\sum_{r=1}^{T} r_t$ , where *T* >> K.




$p(r_t = 1) = \theta_1 \quad \theta_2 \quad \theta_3$

$$Q = \begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccccc} t_0 & t_1 & t_2 & \ldots\ldots & t_m \end{array} \\ \begin{pmatrix} 1 & 0 & \ldots\ldots & 1 \\ 0 & 1 & \ldots\ldots & \vdots \\ 0 & 1 & \ldots\ldots & 0 \end{pmatrix} \end{array}$$

# Modelling the Process as a Beta-Bernoulli bandit with Prior Knowledge

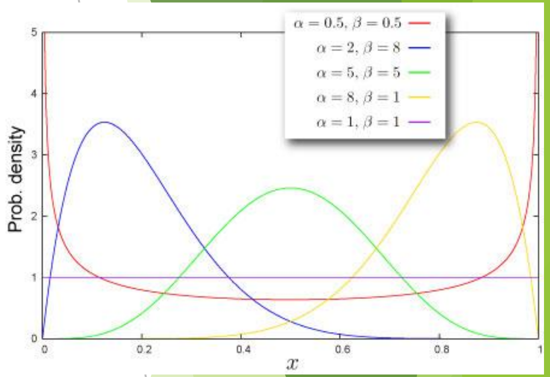
- *LO* = {$lo_1$, $lo_2$, …, $lo_K$}.
- At the $r^{th}$ question of a quiz, reward $x_r \in \{0, 1\}$.
- Take priors to be beta-distributed with parameters $\alpha = (\alpha_1, \dots, \alpha_K)$ and $\beta = (\beta_1, \dots, \beta_K)$.
  - $\alpha_k$ *and* $\beta_k$ correspond to the counts when we **succeeded** or **failed** in learning objective $lo_k$ to get a reward, respectively.
- Each learning objective *k* corresponds to an unknown success probability $\mu_k$ :
  - $p(x_r = 1 | r; lo_k) = \mu_k$, k ∈ {1, 2, …, K}.
- The prior probability density function of $\mu_k$ is

$$p(\mu_k) = \frac{\Gamma(\alpha_k+\beta_k)}{\Gamma(\alpha_k)\Gamma(\beta_k)} \mu_k^{\alpha_k - 1}(1 - \mu_k)^{\beta_k - 1},$$

where $\Gamma$ denotes the gamma function.

- The optimal policy is to choose a question on one learning objective for which $\mu_k$ attains its **smallest** value, i.e.

$$lo^* = argmin_{k \in K} \mu_k.$$




https://ecstep.com/beta-function/

# TS-based Algorithm

- The success probability estimate $\hat{\mu}_k$ is randomly sampled from the posterior distribution, which is a beta distribution with parameters $\alpha_k$ and $\beta_k$, rather than taken to be the expectation $\alpha_k/(\alpha_k+\beta_k)$ used in the greedy algorithm.
- $\hat{\mu}_k$ represents a statistically plausible success probability.

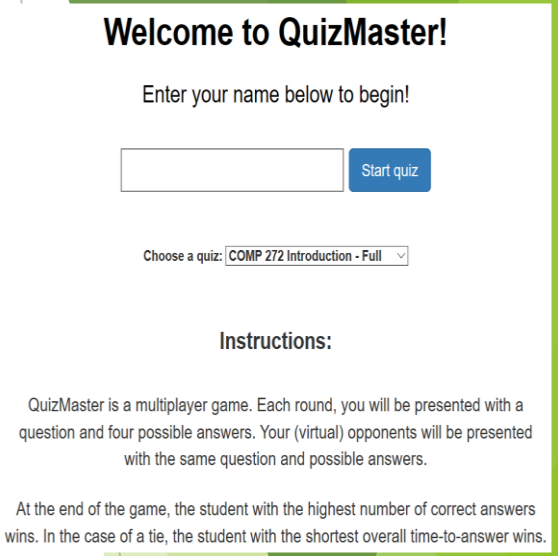
**Algorithm** BernTS-AdaptiveQuizGeneration($LO, \alpha, \beta$)

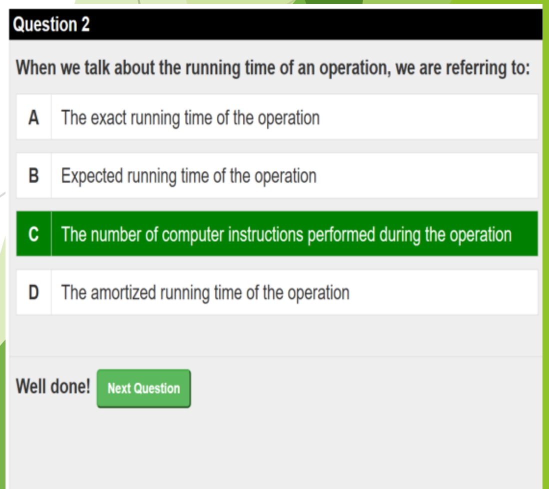
**for** $t = 1, 2, \ldots,$ **do**
    # sample model
    **for** $k = 1, \ldots, K$ **do**
        Sample $\hat{\mu}_k \sim beta(\alpha_k, \beta_k)$
    **end for**
    #select and apply action:
    $lo_t \leftarrow argmin_k \hat{\mu}_k$
    Select a question with $lo_t$ and observes $x_t$
    #update distribution
    $(\alpha_{lo_t}, \beta_{lo_t}) \leftarrow (\alpha_{lo_t} + x_t, \beta_{lo_t} + 1 - x_t)$
**end for**

# Implementation and Experimental Design

- We organize the formative assessment system for a course as several stages, each of which corresponds to a knowledge unit.
- Testing course, Data Structure and Algorithms, having
  - 12 KUs
  - Each LO has at least 3 questions
  - 120 undergraduate students

| | |
|---|---|
| δ1 | Introduction |
| δ2 | Array-based lists |
| δ3 | Linked lists |
| δ4 | Hash Tables |
| δ5 | Recursion |
| δ6 | Binary Trees |
| δ7 | Scapegoat Trees |
| δ8 | Red Black Trees |
| δ9 | Heaps |
| δ10 | Sorting |
| δ11 | Grapes |
| δ12 | External Memory Searching |



# Future Work

- TS-based adaptive quiz generation algorithm
  - Bayesian approach
  - Maximizing the accuracy of identifying lacking areas
  - Prior knowledge
- Data Structure and Algorithms as a testbed
  - Initial stage
  - Deploying and testing
- Benchmarking
  - Positive predictive value (PPV)

# Thank You!