The Effects of Adaptive Learning in a Massive Open Online Course on Learners' Skill Development

Yigal Rosen
ACTNext and Harvard University
Cambridge, USA
yigal_rosen@harvard.edu

Ilia Rushkin
Harvard University
Cambridge, USA
ilia_rushkin@harvard.edu

Rob Rubin
Independent
Sharon, USA
rvrubin@gmail.com

Liberty Munson
Microsoft
Redmond, USA
liberty.munson@microsoft.com

Andrew Ang
Harvard University
Cambridge, USA
andrew_ang@harvard.edu

Gregory Weber
Microsoft
Redmond, USA
gregory.weber@microsoft.com

Glenn Lopez
Harvard University
Cambridge, USA
glenn_lopez@harvard.edu

Dustin Tingley
Harvard University
Cambridge, USA
dtingley@gov.harvard.edu

ABSTRACT

We report an experimental implementation of adaptive learning functionality in a self-paced Microsoft MOOC (massive open online course) on edX. In a personalized adaptive system, the learner's progress toward clearly defined goals is continually assessed, the assessment occurs when a student is ready to demonstrate competency, and supporting materials are tailored to the needs of each learner. Despite the promise of adaptive personalized learning, there is a lack of evidence-based instructional design, transparency in many of the models and algorithms used to provide adaptive technology or a framework for rapid experimentation with different models. ALOSI (Adaptive Learning Open Source Initiative) provides open source adaptive learning technology and a common framework to measure learning gains and learner behavior. This study explored the effects of two different strategies for adaptive learning and assessment: Learners were randomly assigned to three groups. In the first adaptive group ALOSI prioritized a strategy of remediation – serving learners items on topics with the least evidence of mastery; in the second adaptive group ALOSI prioritized a strategy of continuity – that is learners would be more likely served items on similar topic in a sequence until mastery is demonstrated. The control group followed the pathways of the course as set out by the instructional designer, with no adaptive algorithms. We found that the implemented adaptivity in assessment, with emphasis on remediation is associated with a substantial increase in learning gains, while producing no big effect on the drop-out. Further research is needed to confirm these findings and explore additional possible effects and implications to course design.

Keywords

Adaptivity, personalization, assessment, MOOC, edX.

L@S 2018, June 26–28, 2018, London, United Kingdom

ACM ISBN 978-1-4503-5886-6/18/06…$15.00
https://doi.org/10.1145/3231644.3231651

1. INTRODUCTION

Digital learning systems are considered adaptive when they can dynamically change the presentation of content to any user based on the user's individual record of interactions, as opposed to simply sending users into different versions of the course based on preexisting information such as user's demographic information, education level, or a test score. Conceptually, an adaptive learning system is a combination of two parts: an algorithm to dynamically assess each user's current profile (the current state of knowledge, but potentially also affective factors, such as frustration level), and, based on this, a recommendation engine to decide what the user should see next. In this way, the system seeks to optimize individual user experience, based on each user's prior actions, but also based on the actions of other users (e.g. to identify the course items that many others have found most useful in similar circumstances). Adaptive technologies build on decades of research in intelligent tutoring systems, psychometrics, cognitive learning theory and data science [2, 4, 10]. More specifically, Cognitive Tutors utilize knowledge tracing [9] to track knowledge acquisition and provide tailored instruction, by tracking performance on individual production rules in a cognitive model [3, 4, 11]. Extensions to this model have included estimating of the initial probability that the student knows a skill [5], estimating of the impact of help features on probability of acquisition [1], and integrating with models of item difficulty [6]. However, these approaches typically do not consider pacing and require significant content design workload in order to create learning and assessment content [2]. These limitations are critical in large-scale MOOC context. Pioneer studies on adaptive technologies in MOOCs indicated both technical feasibility and the educational promise [7, 8, 9]. Despite the promise of adaptive learning, there is a lack of evidence-based instructional design, transparency in many of the models and algorithms used to provide adaptive technology or a framework for rapid experimentation with different models. Harvard University partnered with Microsoft Learning to develop ALOSI (Adaptive Learning Open Source Initiative) provides open source adaptive learning technology and a common framework to measure learning gains and learner

behavior. The key insights gained from the modeling and analysis work enable us to address the development of evidence-based guidelines for instructional design of future courses, and provides insights into our understanding of how people learn effectively. ALOSI uses Bayesian Knowledge tracing to both develop a predictive model of skills mastery for the learner, and improve the predictive attributes associated with the content. The key features in ALOSI's current adaptive framework include knowledge tracing and recommendation engine, while user modeling, feedback and recommendation of targeted learning materials are in development. The engine improves over time from the use of additional learner data and provides direct insights into the optimization processes (by contrast with commonly used commercial "black box" adaptive engines). Additionally, the architecture of the adaptive engine enables rapid experimentation with different recommendation strategies. This pilot study measured the effects of adaptive pathways on learning gains and dropout rates using different tuning parameters in the adaptive engine against the instructional design learning experience.

2. ALOSI ARCHITECTURE

In order to operationalize ALOSI framework, we developed the Bridge for Adaptivity and the adaptive engine, two open source applications supporting a modular framework for implementing adaptive learning and experimentation that integrates several components: the Bridge for Adaptivity, an Adaptive Engine (such as the ALOSI adaptive engine), a Learning Management System (Learning Tools Interoperability - LTI consumer such as Canvas or edX), and a Content Source (for example, an LTI provider like Open edX). The Bridge for Adaptivity handles the integration of all system components to provide the adaptive learning experience, while the Adaptive Engine provides the adaptive strategy and is designed to be swapped in and out with compatible engines for experimentation and comparison. The diagram in Figure 1 describes the data passing in the system.

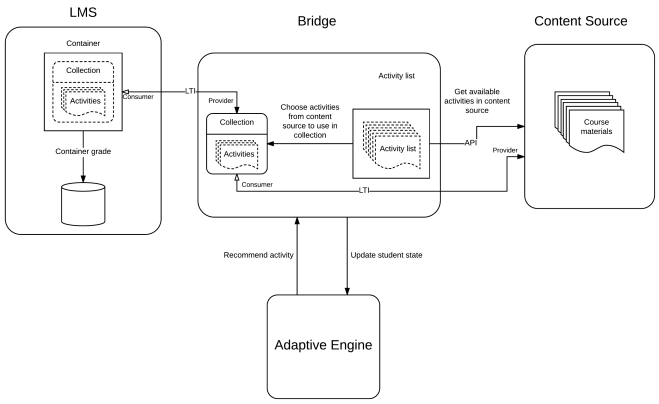

Figure 1. ALOSI Architecture

In this study, the Bridge for Adaptivity was used with the ALOSI adaptive engine to adaptively serve assessments from an Open edX platform instance in the Microsoft MOOC on edX.

The user interface seen by a learner when they encounter an installed tool instance is that shown in Figure 2. Assessment items (problems) from the edX course are displayed one at a time in a center activity window, with a surrounding toolbar that provides features such as navigation, and a score display. Every problem-checking event by the user sends the data to the adaptive engine, to update the mastery information real-time. Every "Next Question" event in an adaptive assessment sends to the engine a request for the next content item to be served to the user (this could a learning or an assessment content). The engine sends back the recommendation, which is accessed as an edX XBlock and loaded.

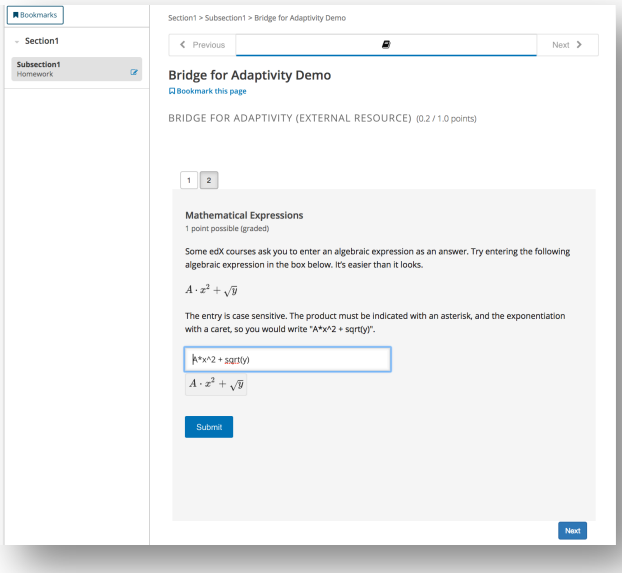

Figure 2. Adaptive assessment user interface

2.1 Adaptive Engine Details

Our goal was to create a simple adaptive recommendation engine for an edX MOOC, capable of deciding what item to serve to a user next based on the user's history. We use a variety of Bayesian Knowledge Tracing (BKT) model to estimate the students' state. What makes our situation special is that, as we learned from the adaptive pilot and just generally from seeing MOOCs,

1) Questions in the course differ widely in nature, and in particular in difficulty. Thus, we cannot assign the same values of guess, slip and transit probabilities to them, even if they are all tagged with the same knowledge component.

2) Tagging is complicated: some of the questions are tagged with multiple knowledge components.

3) In a self-paced MOOC environment, there is a need for a causal structure in the knowledge components: we should not serve to a user items tagged with a knowledge component, if the user has shown lack of knowledge of other knowledge components that are pre-requisite to that one. In the simplest case, it can be dictated by a simple ordered list (the natural order of learning the content of the course), but it could also be a detailed graph of pre-requisite relationships among knowledge components.

4) In a MOOC, the number of students is high, so we can afford to define a model with a large number of parameters and optimize them based on the student interaction data.

Conceptually, our engine consists of two blocks: knowledge tracing and the recommendation engine, which uses the output of knowledge tracing as an input.

Knowledge Tracing

Let there be Q questions in the course ($q = 1,2 \dots Q$), tagged with N knowledge components ($i = 1,2 \dots N$), or KCs for short. We introduce matrices of guess, slip and transfer probabilities of the questions: $p_{qi}^{\text{guess}}, p_{qi}^{\text{slip}}, p_{qi}^{\text{trans}}$, which are the generalizations of the usual guess slip and transfer parameters of BKT [6]. We do not assume these parameters to the same for all questions, due to the item diversity.

We assume that the mastery of each KC by each course user is a binary latent variable – the user either has learned it or not – and we update the mastery matrix p, where the element p_{ui} is the currently estimated probability that the user u has the mastery of the KC i. We define the mastery threshold $p^* \in [0,1]$, and if $p_{ui} \geq p^*$, we say that the mastery of i by the user u is sufficiently certain and no longer needs verification. We initialize the mastery probability matrix $p = p^{(0)}$ (user' prior knowledge), after which, when a user submits an answer to the question, it gets a correctness value (score) $C_q^{(u)} \in [0,1]$ and we update the mastery probability of each KC (i.e. this user's row of the matrix p). We also characterize

The Bayesian updating is easier to write in terms of odds, or even logarithmic odds, rather than the probability p:

$$\mathcal{O}_{ui} = \frac{p_{ui}}{1 - p_{ui}}, \qquad L_{ui} = \log \mathcal{O}_{ui}, \qquad L^* = \log \frac{p^*}{1 - p^*} \qquad (1.)$$

So we will translate the transit, guess and slip probabilities into odds as well: $o_{qi}^{\text{guess}} = p_{qi}^{\text{guess}} / (1 - p_{qi}^{\text{guess}})$ etc, and introduce the likelihood ratios for the case of incorrect (0) and correct (1) answer:

$$x_{qi}^0 = \frac{p_{qi}^{\text{slip}}}{1 - p_{qi}^{\text{guess}}}, \qquad x_{qi}^1 = \frac{1 - p_{qi}^{\text{slip}}}{p_{qi}^{\text{guess}}} \qquad (2.)$$

These matrices encode the *relevance* of a question q to a KC i. If the problem is irrelevant to a KC, the probability of correct or incorrect score should be independent of that KC. This will be the case if $p_{qi}^{\text{slip}} = 1 - p_{qi}^{\text{guess}}$, in which case $x_{qi}^0 = x_{qi}^1 = 1$. We propose to define the *relevance* matrix, which is essentially a generalization of tagging, as a sum of logarithmic odds of non-guessing and non-slipping:

$$k_{qi} = \log x_{qi}^1 - \log x_{qi}^0 = -\log o_{qi}^{\text{guess}} - \log o_{qi}^{\text{slip}} \qquad (3.)$$

This can be viewed as a generalization of tagging items with KCs. While the tagging matrix is binary (a KC is either linked to a question or not), the relevance matrix shows the weight of each link: how much of an evidence for the KC mastery the question provides. The multiplicative factor earned by the mastery odds is:

$$x_{qi} = x_{qi}^0 \left(\frac{x_{qi}^1}{x_{qi}^0} \right)^{C_q^{(u)}} \qquad (4.)$$

For binary (0 or 1) scores, this is just another way of saying that the factor should equal x_{qi}^0 or x_{qi}^1. But we can also interpolate for fractional scores, and this is what Eq. 4 does. Exactly how we interpolate between these for fractional scores is a matter of choice. For instance, an alternative definition could be a linear interpolation $x_{qi} = x_{qi}^0 + C_q^{(u)} (x_{qi}^1 - x_{qi}^0)$. We settled on the multiplicative interpolation by looking at the location of the "borderline" score, for which $x_{qi} = 1$, reresenting the boundary between correctness and incorrectness. For instance, as a back-of-the-envelope estimate, let the guess and slip probabilities have equal values (typically, they are not too different). In Eq. 1, this sets the borderline score at a reasonable 0.5, whereas in case of linear interpolation the borderline score in such a situation equals the slip (= guess) probability, which is likely too low.

The posterior odds, with the evidence of the submitted problem, become $\mathcal{O}_{ui} \to \mathcal{O}_{ui} x_{qi}$. Additionally, we modify the mastery odds due to transfer of knowledge, so the full update procedure is:

$$\mathcal{O}_{ui} \to o_{qi}^{\text{trans}} + \left(o_{qi}^{\text{trans}} + 1 \right) \mathcal{O}_{ui} x_{qi} \qquad (5.)$$

This is a type of Bayesian Knowledge Tracing. The main modification is that we deliberately formulated it that we formulate it in such a way that there is no explicit requirement to tag each question with only one KC. If a problem is tagged with several KCs ($k_{qi} > 0$ for more than one value i. We essentially view the problem as a collection of sub-problems, each tagged with a single KC. This is our proposed the generalization of BKT to multiple tagging. The predicted odds of correct answer are found as

$$\mathcal{O}_{qu}^{\text{pred}} = \prod_i \frac{\mathcal{O}_{ui} \left(1 - p_{qi}^{\text{slip}} \right) + p_{qi}^{\text{guess}}}{\mathcal{O}_{ui} p_{qi}^{\text{slip}} + 1 - p_{qi}^{\text{guess}}} \qquad (6.)$$

which is to say that we take the ratio of the probability that each sub-problem is answered correctly to the probability that each sub-problem is answered incorrectly (since we must remove from the ensemble the possibilities of correct answer on some but not all sub-problems).

The outlined procedure is multiplicative in nature. An obvious idea would be to replace it with an additive one by working with logarithmic odds L_{ui} (which we do, in fact, in the recommendation part of the engine). It would be clearly preferable from the computational point of view in the knowledge-tracing part as well, if it was not for the knowledge-transfer step: in the additive formulation this step would involve an exponentiation and a taking a logarithm.

For terminological simplicity we referred to the content items as questions. However, the model can accommodate instructional items as well, e.g. videos or text. We can adopt a rule that, if an item q is instructional, the outcome of user's interaction with it is always "correct". A way to think of it is to imagine that q includes an assessment part of trivial difficulty. The slip probabilities $p_{qi}^{\text{slip}} = 0$, the guess probabilities now have the meaning of the probability of not learning an KC from the item, and so we set them to $p_{qi}^{\text{guess}} = 1 - p_{qi}^{\text{trans}}$.

If the matrix p or other parameter matrices contain zeros or ones it is possible to encounter 0/0 indeterminacies. One way to preclude these is adopt a small cutoff, e.g. we can set $\epsilon = 10^{-10}$, and coerce all elements of the parameter matrices $p^{\text{slip}}, p^{\text{guess}}, p^{\text{trans}}$, as well as the initial mastery probability $p^{(0)}$, to the interval $[\epsilon, 1 - \epsilon]$.

Learning parameters of knowledge tracing

We will rely on a way to optimize our BKT parameters, inspired by the "empirical probabilities" method of [7].

At regular points in time, when we decide to run the optimization, suppose that the items submitted by a user u are $\{q_j^{(u)}\}$ ($j = 1,\dots J^{(u)}$), indexed in chronological order, and let the correctness scores be $C_j^{(u)}$. We denote $K_{ij}^{(u)}$ this student's latent mastery of a KC i just before submitting the item $q_j^{(u)}$. Assuming that there is no forgetting, the knowledge is a non-decreasing function with values 0 and 1, so it is characterized simply by the position of the unit step: for j from 1 to some n_i knowledge is 0 and from there onward it is 1. We need to find which n_i gives the highest accuracy of predicting correctness from knowledge. Once this is done, the knowledge is not a latent variable anymore, and we can estimate guess, slip and transfer probabilities by frequencies of observations.

The generalized number of errors on predicting the outcome based on mastery of a particular knowledge component are:

$$E_i^{(u)}(n) = -\sum_{j=1}^{n} C_j^{(u)} \log o_{q_j i}^{\text{guess}} - \sum_{j=n+1}^{J^{(u)}} (1 - C_j^{(u)}) \log o_{q_j i}^{\text{slip}} \quad (7.)$$

where $n \in [0, J^{(u)}]$ and we adopt the convention that if the lower limit of a sum is greater than the upper limit, the sum is 0. We set the knowledge step location for each KC: $n_i = \text{argmin}(E_i^{(u)})$, and construct the step-function $K_{ij}^{(u)}$ using it. If there are multiple equal minima, and hence multiple n_i, we take the average of the corresponding multiple step-functions (because of this, knowledge may now have fractional value). Note that, if user's problems are irrelevant for an KC, we will find a steadily growing knowledge of that KC. This is not bad, however, since for each KC we will average only over the users who experienced some relevant problems. Namely, we can define the sets of users

$$\mathcal{U}_i = \{\forall u: \sum_{j=1}^{J^{(u)}} k_{q_j^{(u)} i} > \eta\}$$

$$\mathcal{U}_{qi} = \{\forall u: \sum_{j=1}^{J^{(u)}} k_{q_j^{(u)} i} \mathbf{1}(q_j^{(u)} = q) > \eta\}$$

where $\eta \geq 0$ is a constant we set as a measure of how much total relevance of a KC is enough for the user to be included into the ensemble for estimating the parameters of that KC. As the simplest choice, in this implementation we set $\eta = 0$.

Now we can estimate the BKT parameter matrices from the user data:

$$p_{u'i}^{(0)} = \frac{\sum_{u \in \mathcal{U}_i} K_{i1}^{(u)}}{\sum_{u \in \mathcal{U}_i} 1} \quad (8.)$$

(same prior knowledge for all users u').

$$p_{qi}^{\text{guess}} = \frac{\sum_{u \in \mathcal{U}_{qi}} \left(\sum_{j=1}^{J^{(u)}} \left(1 - K_{ij}^{(u)}\right) C_j^{(u)} \mathbf{1}(q_j^{(u)} = q) \right)}{\sum_{u \in \mathcal{U}_{qi}} \left(\sum_{j=1}^{J^{(u)}} \left(1 - K_{ij}^{(u)}\right) \mathbf{1}(q_j^{(u)} = q) \right)} \quad (9.)$$

$$p_{qi}^{\text{slip}} = \frac{\sum_{u \in \mathcal{U}_{qi}} \left(\sum_{j=1}^{J^{(u)}} K_{ij}^{(u)} \left(1 - C_j^{(u)}\right) \mathbf{1}(q_j^{(u)} = q) \right)}{\sum_{u \in \mathcal{U}_{qi}} \left(\sum_{j=1}^{J^{(u)}} K_{ij}^{(u)} \mathbf{1}(q_j^{(u)} = q) \right)} \quad (10.)$$

$$p_{qi}^{\text{trans}} = \frac{\sum_{u \in \mathcal{U}_{qi}} \left(\sum_{j=1}^{J^{(u)}-1} \left(1 - K_{ij}^{(u)}\right) K_{i,j+1}^{(u)} \mathbf{1}(q_j^{(u)} = q) \right)}{\sum_{u \in \mathcal{U}_{qi}} \left(\sum_{j=1}^{J^{(u)}-1} \left(1 - K_{ij}^{(u)}\right) \mathbf{1}(q_j^{(u)} = q) \right)} \quad (11.)$$

Here again, we adopt the convention that if the lower limit of a sum is greater than the upper limit, the sum is 0 (this happens when $J^{(u)}$ is 0 or 1). The value of the denominator in each of these expressions is a measure of how much student information we have for estimating the probability. In case there is no data, the expression becomes a 0/0. We should not want to update a probability in this case. Moreover, we imposed a threshold $M = 20$ and did not update a particular matrix element if the denominator in the corresponding equation is less than M. Likewise, we did not update if the calculated value was degenerate, e.g. a guess probability and a slip probability add up to more than 1.

The updated prior knowledge values $p^{(0)}$ will be used for all users yet to come to the course, but also for the existing users for those knowledge components that they have not yet been exposed to.

Recommendation engine

The strategy we use for recommending the next item is a weighted combination of a number of sub-strategies. Each sub-strategy comes in with an importance weight (the vector of these weights is a governing parameter of the adaptive engine).

Let us first define the matrix of pre-requisite readiness. The pre-requisite relationships among the KCs are naturally visualized as a directed acyclic graph, and are stored as an $N \times N$ matrix w of pre-requisite strengths, w_{ij} representing the strength of the graph edge (KC j is a pre-requisite for KC i). We define this strength to be on the scale from 0 to 1. If the SME provided no pre-requisite relations form the KCs, w a zero matrix.

The pre-requisite readiness is defined for each KC and for each user as a matrix:

$$r_{ui} = \sum_{j=1}^{N} w_{ij} \min\left(0, L_{uj} - L^*\right) \quad (12.)$$

An element r_{ui} has value 0 if the user has sufficiently mastered all KCs pre-requisite for the KC i, and less than 0 if the mastery probabilities for some pre-requisites are not yet certain. If the pre-requisite strength w_{ij} is weaker, it enters r_{ui} with a smaller weight, allowing less certain mastery of less important pre-requisites. If all the pre-requisites are ascertained, $r_{ui} = 0$, otherwise it is negative. We can deviate from this slightly and introduce a forgiveness parameter $r^* \geq 0$, so that a user u is sufficiently ready for learning a KC i if $r_{ui} + r^* \geq 0$.

To recommend the next question for a student, we subset the relevance matrix k_{qi} to only those questions (matrix rows) that belong to the adaptive module where the user u is and that the user has not seen yet. Thus, we obtain a user specific matrix $k_{qi}^{(u)}$. We define the non-negative user-specific vectors of "remediation", "continuity", "difficulty matching", and "readiness" (in terms of difficulty level of the problem $d_q \in [\epsilon, 1 - \epsilon]$):

$$R_q^{(u)} = \sum_{i=1}^{N} k_{qi}^{(u)} \max(0, L^* - L_{ui}) \quad (13.)$$

$$C_q^{(u)} = \sqrt{\sum_{i=1}^{N} k_{qi}^{(u)} k_{q_{\text{last}},i}} \quad (14.)$$

$$D_q^{(u)} = -\sum_{i=1}^{N} k_{qi}^{(u)} \left| L - \log\frac{d_q}{1-d_q} \right| \qquad (15.)$$

$$P_q^{(u)} = \sum_{i=1}^{N} k_{qi}^{(u)} \min(0, r_{ui} + r^*) \qquad (16.)$$

where q_{last} is the last item the user saw.

These expressions formulate the four sub-strategies of our recommendation engine. The vectors are the ratings of all potential items by the sub-strategies. The first sub-strategy, "remediation", rates higher those items on whose KCs the user's mastery is currently low. The second, "continuity", rates higher items tagged most similarly to the last seen item. The third favors items with the difficulty level that matches the mastery level and the fourth tries to avoid serving a question if the user has not mastered the KCs that are pre-requisite to the KCs of that question.

More competing subs-strategies can be added to the list at will, but in this implementation we used these four. We introduce a vector of sub-strategy weights: $W = (W_r, W_c, W_d, W_p)$, defined up to normalization. So that the overall rating of the available items is the weighted sum:

$$S_q^{(u)} = W_r R_q^{(u)} + W_c C_q^{(u)} + W_d D_q^{(u)} + W_p P_q^{(u)} \qquad (17.)$$

The item q that maximizes $S_q^{(u)}$ will be served to the user u.

The serving stops naturally when we exhausted the available questions (the matrix $k^{(u)}$ has no rows). Additionally, we may adopt a "stop on mastery" policy and stop serving if $R_q^{(u)} = 0$ for all q, which means that the user has reached the mastery threshold p^* on all KCs relevant for the available pool of items.

2.2 Method

Adaptive functionality has been deployed in Microsoft MOOC on edX "Essential Statistics for Data Analysis Using Excel". The instructional design team significantly enhanced the assessment scope, and included over 35 knowledge components and 400 assessment items tagged to those knowledge components. Our experimental design randomly assigned learners in the course to three independent groups: in the first adaptive group ALOSI prioritized a strategy of remediation – serving learners items on topics with the least evidence of mastery (Group A); in the second adaptive group ALOSI prioritized a strategy of continuity – that is learners would be more likely served items on similar topic in a sequence until mastery is demonstrated (group B); the control group followed the pathways of the course as set out by the instructional designer, with no adaptive algorithms (Group C). Thus, groups A and B of the students experienced two varieties of the adaptive engine.

The difference was in the recommendation sub-strategy weights. For group A, the weight of remediation was set to 2, and that of continuity to 1. For group B these values were reversed. The weights of the remaining two sub-strategies were the same for both groups: 1 for pre-requisite readiness and 0.5 for difficulty matching. The mastery threshold L* was set to 2.2 (corresponding to p* about 0.9. The pre-requisite forgiveness r* was set to 0. The serving policy "stop on mastery" was not used: as long as a user requested more adaptive questions, they were served until the available pool was exhausted.

Note that the continuity sub-strategy does not use the answer correctness. Therefore, Group B experienced less variability in serving order than Group A (And Group C experience none at all). Furthermore, at the request of the course team, we suspended adaptive serving in the beginning of two assessment modules: the pre-test and the post-test. In these, for Groups A and B, the first 34 or 35 (respectively) items were served in a fixed sequence (same for everyone), and only afterwards the serving order became adaptive.

It should be noted that the approach in the first adaptive group was the most different from the conventional non-adaptive learning experience of the third group, and the second adaptive group occupies the intermediate position. Moreover, in the adaptive groups the learners were working on one item at a time, while in the control group the items were presented in the conventional edX approach – several items at once.

From the course SME we obtained the information about the assessment items: a list of KCs, a list of pre-requisite relations among them, tagging of items with KCs, difficulty level of each item and basic estimates of the guess, slip and transfer probabilities. These were used as cold guesses at the start, and in the progress of the course these values were updated with those learned from the data. The numerical estimates (e.g. the difficulty level or the connection strength between two KCs) were estimated by the SME using a 3-level scale (weak/medium/strong), which we then converted to numbers for the use in the engine.

Although our engine is capable of operating with multiple tagging, in this course it did not happen: each item was tagged with only one KC.

3. Findings

All students in the course were administered a pre-test and a post-test, allowing a comparison of learning gains across three groups of students. For the adaptive groups A and B, the first 34 problems in the pre-test and the first 35 in the post-test were served non-adaptively: their sequence was fixed, and only the remainder of problems in both tests was served adaptively. Thus, we use the average problem score of only these fixed parts of the tests for the comparison, to ensure that all students are compared on equal footing. For Group C we simply use the entire pre-test and post-test that this group received.

We observe no substantial differences across the groups in the average problem score in the pre-test, confirming the assumption that initially the composition of the three groups is comparable[1]. If anything, group A was at a slight disadvantage initially.

The learning gains are observed as the difference between the average problem score in the post-test and in the pre-test. It appears that group A experienced the greatest learning gain (ES=0.641). Group B, whose version of adaptivity was weaker (continuity was emphasized rather than remediation), has lower

[1] Everywhere in this paper, by p value we mean the p-value from the two-tailed t-test, and by the effect size (ES) we mean Cohen's d.

learning gains (ES=0.542), and the control Group C had still less (ES=0.535).

Table 1. Learning gains across the three groups

Pre-test	Group A	Group B	Group C
Pre-test mean score	0.491	0.520	0.510
Post-test mean score	0.782	0.768	0.758
Effect size of learning gains	**0.641**	**0.542**	**0.535**

We estimate standard error of the post-test participation rates with the help of binomial distribution as slightly over 1% in all three groups, which means that the differences between the post-test participation are insignificant.

In the learning gains analysis above we included all users who submitted at least one question in a pre-test and in a post-test, i.e. students who are both pre-testers and post-testers. So the question remains how many of the pre-testers dropped out without reaching the post-test.

We further investigate the effect of the experimental groups on learning gains: how much of it was due to the simple fact that experimental users had access to many more questions in the learning modules than the control users, and therefore had more chances to practice their knowledge? The number of questions in the fixed sequences in the pre-test and post-test for the experimental groups was 34 and 35, respectively. The number of questions in the pre-test and the post-test for the Control group was 29 and 30 respectively. We have 793 (Remediation/Continuity/Control=238/263/292) users who submitted at least one question in the pre-test and at least one question in the post-test, but restricting the analysis to those who submitted the minimum of 29 pre-test and 30 post-test questions (the numbers of questions from the Control group). As a result, the number of users left is 448 (Remediation/Continuity/Control=127/154/167). Defining the learning gain as the difference between a user's post-test mean score and pre-test mean score, we train on these users a linear model where the outcome is the learning gain and the explanatory variables are the pre-test mean score, the experimental group, and the number of questions submitted in the modules 1-5 of the course. The adjusted R-squared of the model is 0.24. As a result, belonging to group A ("remediation") increases the gain by 0.057 (p=0.03) compared to the control group C; belonging to group B ("continuity") has no significant effect (p=0.54). Furthermore, the number of problems turns out to have no statistically significant effect on the learning gain (p=0.65), suggesting that the benefit of remediation adaptivity is not explained as simply the benefit of practicing with more questions.

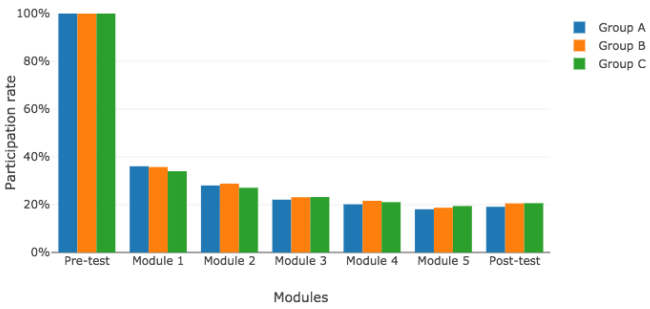

Figure 3. Participation rates by course module

In Figure 3, only the pre-testers are included, so the participation in the pre-test is by definition 100% in any group. The biggest drop-out occurs early on in the course, which is typical for any MOOC. Also, there is a small number of learners who skip the assessment in some modules but go to the post-test - this is manifest from the fact that the participation rate in the post-test is higher than in module 5. The numbers of learners in this graph are A/B/C=1245/1281/1415. The participation rates in the post-test are A/B/C=19.1/20.5/20.6% We estimate standard error of the post-test participation rates with the help of binomial distribution as slightly over 1% in all three groups, which means that the differences between the post-test participation are insignificant.

We conclude that the implemented adaptivity in assessment with emphasis on remediation (Group A) is associated with a substantial increase in learning gains, while producing no big effect on the drop-out.

The knowledge tracing, which occurs in our engine, allows determining the demonstrated mastery probability for any knowledge component and for any learner after any submit event. This opens up the possibility of visualizing the learning curve, rather than simply relying on the difference between pre-test and post-test scores. Given that we have so many knowledge components, we prefer to aggregate them in groups for the purpose of visualization. Our approach is as follows. Within any assessment module, we average the mastery probabilities of any user across all the knowledge components that are represented in the tagging of the problems in that module. In this way, we create for each user an overall mastery level in a module. Then we can consider group averages of this overall mastery level. In the figure below we plot these group averages of mastery vs. the number of problems tried by a user in the module.

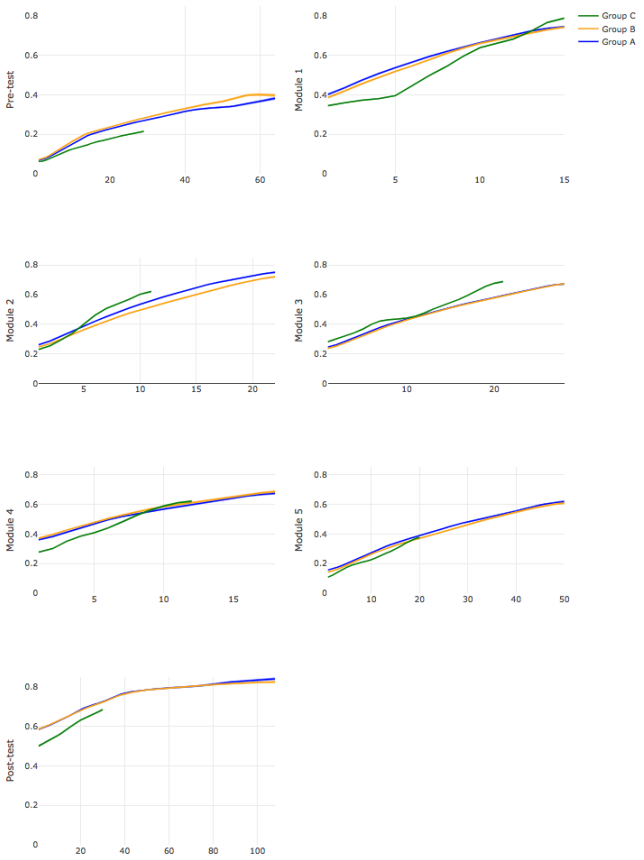

Figure 4. Learner curves by Group, by course module

One noticeable feature is that in many assessment modules the learning curves of adaptive groups are smoother. As the plots show, group C often had a smaller item bank than the adaptive groups, and with the exception of the pre-test, almost all users in this group submitted almost all problems (in the table below we show the mean percentages of submitted problems).

Table 1. Average percentage of problems submitted

	Average percentage of problems submitted						
	Pre-test	Module 1	Module 2	Module 3	Module 4	Module 5	Post-test
Group A	26%	86%	85%	86%	90%	72%	60%
Group B	26%	87%	88%	87%	90%	71%	63%
Group C	51%	93%	96%	97%	95%	93%	91%

Therefore, the sharp twists in the Group C learning curves are not explained away by population stratification. Adaptivity produces a smoother learning experience.

4. CONTRIBUTIONS

Our experimentation with adaptive assessments provided initial evidence on the effects of adaptivity in MOOCs on learning gains and dropout rates. Furthermore, the architecture of the Bridge for Adaptivity and the adaptive engine developed in this project enables rapid experimentation with different recommendation strategies in the future. In this study, adaptivity was implemented on Multiple-Choice assessment problems. There appear to be extensive opportunities to expand adaptive engine to a broad range of assessment item types and enable adaptivity in learning content (e.g., videos, readings, simulations) in MOOCs. Given the structure of many MOOCs, more integration between learning content and assessment could provide an adaptive experience that would guide learners to content that could improve their understanding based on how they perform on integrated assessments. Additional factors could be included to provide a more personalized learning experience. We can conceive an adaptive engine that decides what item to serve next based not just on the mastery but also on career interests and behavioral patterns interpreted as boredom or frustration.

In addition, we anticipate expanding this adaptive learning system to work with other LTI-compliant Learning Management Systems on a large scale.

5. ACKNOWLEDGMENTS

We are grateful for the support from the Office of the Vice Provost for Advances in Learning at Harvard University and Microsoft.

6. REFERENCES

[1] Beck, J., Chang, K-M., Mostow, J., and Corbett, A. 2008. Does help help? Introducing the Bayesian Evaluation and Assessment methodology. In *International Conference on Intelligent Tutoring Systems*. Springer, 383–394

[2] Hawkins, W.J., Heffernan, N.T. and Baker, R.S., 2014. Learning Bayesian knowledge tracing parameters with a knowledge heuristic and empirical probabilities. In *International Conference on Intelligent Tutoring Systems* (pp. 150-155). Springer, Cham.

[3] Koedinger, K., Anderson, J., Hadley, W., and Mark, M. 1997. *Intelligent tutoring goes to school in the big city*. (1997).

[4] Koedinger, K., and Stamper, J. 2010. A Data Driven Approach to the Discovery of Better Cognitive Models. In Baker, R.S.J.d., Merceron, A., Pavlik, P.I. Jr. (Eds.) *Proceedings of the 3rd International Conference on Educational Data Mining*. (EDM 2010), 325-326. Pittsburgh, PA.

[5] Pardos, Z., and Heffernan, N. 2010. Navigating the parameter space of Bayesian Knowledge Tracing models: Visualizations of the convergence of the Expectation Maximization algorithm. In Educational Data Mining 2010.

[6] Pardos, Z., and Heffernan, N. 2011. KT-IDEM: Introducing item difficulty to the knowledge tracing model. In *International Conference on User Modeling, Adaptation, and Personalization*. Springer, 243–254.

[7] Pardos, Z., Tang, S., Davis, D., and Vu Le C. 2017. *Enabling real-time adaptivity in MOOCs with a personalized next-step recommendation framework*. Proceedings of the Fourth ACM Conference on Learning @ Scale.

[8] Rosen, Y., Rushkin, I., Ang, A., Federicks, C., Tingley, D., and Blink, M. - J. 2017. *Designing adaptive assessments in MOOCs*. Proceedings of the Fourth ACM Conference on Learning @ Scale.

[9] Rushkin, I., Rosen, Y., Ang, A., Fredericks, C., Tingley, D., Blink, M. J., and Lopez, G. 2017. *Adaptive Assessment Experiment in a HarvardX MOOC*. Proceedings of the 10th International Conference on Educational Data Mining.

[10] Stamper, J., Barnes, T., and Croy, M. 2011. Experimental Evaluation of Automatic Hint Generation for a Logic Tutor. In Kay, J., Bull, S. and Biswas, G. eds. *Proceeding of the 15th International Conference on Artificial Intelligence in Education (AIED2011)*. 345-352. Berlin Germany: Springer.

[11] Corbett, A. T. and Anderson, J. R., 1994. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, *4*(4), 253-278.