Interactive Learning Environments

ISSN: 1049-4820 (Print) 1744-5191 (Online) Journal homepage: http://www.tandfonline.com/loi/nile20

Adaptively selecting biology questions generated from a semantic network

Lishan Zhang & Kurt VanLehn

To cite this article: Lishan Zhang & Kurt VanLehn (2016): Adaptively selecting biology questions generated from a semantic network, Interactive Learning Environments, DOI: 10.1080/10494820.2016.1190939

To link to this article: http://dx.doi.org/10.1080/10494820.2016.1190939

Published online: 16 Jun 2016.

INTERACTIVE LEARNING ENVIRONMENTS, 2016
http://dx.doi.org/10.1080/10494820.2016.1190939

Routledge
Taylor & Francis Group

Adaptively selecting biology questions generated from a semantic network

Lishan Zhang and Kurt VanLehn

CIDSE, Arizona State University, Tempe, AZ, USA

ABSTRACT

The paper describes a biology tutoring system with adaptive question selection. Questions were selected for presentation to the student based on their utilities, which were estimated from the chance that the student's competence would increase if the questions were asked. Competence was represented by the probability of mastery of a set of biology knowledge components. Tasks were represented and selected based on which knowledge components they addressed. Unlike earlier work, where the knowledge components and their relationships to the questions were defined by domain experts, this project demonstrated that the knowledge components, questions and their relationships could all be generated from a semantic network. An experiment found that students using our adaptive question selection had reliably larger learning gains than students who received questions in a mal-adaptive order.

ARTICLE HISTORY
Received 16 December 2015
Accepted 14 May 2016

KEYWORDS
Adaptive learning; question generation; student modeling; adaptive test items selection; Bayesian Knowledge Tracing

1. Introduction

Practicing is necessary for learning and especially for retention. Thus, a typical textbook chapter has exercises at its end that allow students to practice using the knowledge conveyed in the chapter. A typical math, physics or engineering textbook may have a hundred or more problems at the end of each chapter. If the textbook has an online homework website, then it typically can generate an infinite number of problems by inserting randomly generated numbers into templates.

However, other subjects are not as fortunate as the mathematically intensive ones. For instance, a typical college biology textbook has only a handful of questions at the end of each chapter. Online resources provide a few more questions for students to answer, but the questions tend to duplicate those in the chapters and there are not enough of them. Thus, when students are trying to practice using knowledge for a non-mathematical subject, they rapidly run out of questions to answer and must resort to re-reading the chapter.

Our overarching goal is to develop technologies for web-based practice systems in biology and other non-mathematical subjects. Such systems should have at least five properties:

(1) They should be able to generate an essentially infinite number of practice questions so that students will have ample opportunities to practice using the knowledge they have acquired while listening to lectures or reading the textbook.
(2) The systems should also be adaptive, in that they select or recommend questions that will maximize the student's learning and engagement.
(3) The systems should provide helpful corrective feedback when students answer incorrectly.

CONTACT Lishan Zhang lishan.zhang3@gmail.com

(4) The systems should ask both open response questions, where students type in an answer, and multiple-choice questions, where the answer is entered more rapidly, with a single click.
(5) The multiple-choice questions should offer plausible incorrect answers, so that students with inchoate misconceptions may be enticed to articulate them and get corrective feedback on them.

In earlier work (Zhang, & VanLehn, 2016), we compared two techniques for generating biology questions automatically. One was based on mining the web. The other was based on reasoning from a biology knowledge base that others had generated for answering (not asking) biology questions. Human judges compared questions from textbooks to both the web-mined questions and the inferred questions. The judges rated the questions for depth, pedagogical effectiveness, fluency and ambiguity. Analyses of the ratings found that on most dimensions, the inferred questions were just as good as the textbook questions and web-mined questions, but they tended to be shallow, factual questions. An attempt to generate deep questions from the knowledge base foundered, and that issue remains a topic for future work. However, shallow questions are quite important in biology and similar domains, since it is likely that students will unable to learn much from attempting to answer deep questions until they have first mastered the knowledge required by shallow questions.

This paper reports our progress on the second required feature of a biology practice website: adaptive selection of questions. Practice can be a boring waste of time when the questions exercise only knowledge that the student has mastered. On the other hand, practice can be a frustrating waste of time when the student is not yet prepared to learn from the question. Because different students have different prior knowledge and learning rates, a practice system should provide personalized selection of practice questions.

However, in order to select a question that matches the student's current state of knowledge, the system must formally represent both the student's current state of knowledge and the knowledge exercised by the question. That is, the system needs two models. It needs a *student model* that indicates the profile of competencies of each student and a *question model* that indicates the knowledge is required by each question. A simple student model is a matrix whose rows represent students and columns represent pieces of knowledge; a cell has a number indicating the competence of that student on that piece of knowledge. A *Q-matrix* (Tatsuoka, 1996) is a simple type of question model. Its rows represent questions and its columns represent pieces of knowledge; a cell is 1 if that question's answer requires that piece of knowledge, and 0 otherwise.

The q-matrix and the student model share a decomposition of the domain knowledge into pieces. The pieces have many names; we will use "knowledge component". Other common names are skills and factors.

The Q-matrix is typically created by human authors and not revised from data. In particular, as human experts author a question or question template, they also specify the knowledge components required for answering it. This can be tedious and inaccurate. When the q-matrix is done, its columns specify a decomposition of the domain knowledge into knowledge components. This decomposition allows an empty student model to be built. The student model is initialized with estimates of the students' competence and updated every time a student responds to a question. There are many methods for doing such updates, and some will be reviewed later.

Our contribution is showing that when the questions are generated from a knowledge base, the knowledge base can also be used to generate the q-matrix and thus define the knowledge components for the student model. This is not technically challenging, but it is risky in the sense that the resulting models might not align well with student's actual knowledge structures, which could cause adaptive question selection to behave poorly.

Thus, we tested the resulting adaptive question selection system to see if the adaptation had the desired impact on learning gains. We compared the learning of students who answered questions selected to *increase* their learning gains to students given questions selected to *decrease* learning gains. As expected, the manipulation did influence learning gains, and in the expected direction.

This shows that learning gains are sensitive to adaptive question selection, which was the major risk factor. To establish whether the effect is large enough to be useful would require another experiment comparing adaptive question selection to some baseline treatment, such as presenting questions in fixed order.

The next section reviews relevant prior work on adaptive task selection. In our application, a task is answering a biology question. The subsequent section discusses the technology, focusing on the technique used to understand the students' typed responses to questions and on the Bayesian methods used for updating the student model and selecting questions. The last half of the paper presents the experiment used to test the viability of the approach.

2. Prior work on adaptive task selection

Many practice systems on the market and in the literature are adaptive in that they make decisions based on the student's performance so far. Although this review focuses on systems that select tasks for the user (in biology, a question is a task), it is worth mentioning that some systems make other types of decisions, such as: Which hint to start with in a sequence of increasingly strong hints? (Wood, 2001) Whether to give an unsolicited hint or to respond with a hint when asked for one? (Beck, Woolf, & Beal, 2000; Murray & VanLehn, 2006; Murray, VanLehn, & Mostow, 2004) Which behavior an affective agent should perform? (Arroyo, Mehranian, & Woolf, 2010; Muñoz, Mc Kevitt, Lunney, Noguez, & Neri, 2010) Should the task be posed as an example, a problem with feedback or a problem without feedback? (Rafferty, Brunskill, Griffiths, & Shafto, 2011; Whitehill, 2012)

Many systems are adaptive in that they implement mastery learning (Bloom, 1974). With mastery learning, tasks are partitioned into sets such that each set of tasks addresses a single instructional objective. Students are repeatedly assigned tasks from the same set until their performance rises about a certain level. At this point, they are said to have mastered the instructional objective and they move to another instructional objective and its set of tasks. We are not addressing mastery learning here, as there has been considerable research on it already (Guskey & Gates, 1986; Kulik, Kulik, & Bangert-Drowns, 1990; Slavin, 1990). Whereas mastery learning is concerned with advancing from one set of candidate tasks to another set, we are concerned with selecting a task from the set of candidates.

In a sense, the mathematically optimal way to select tasks is to use a Partially Observable Markov Decision Process (POMDP). The basic idea is to predict what the students' state will be after one, two or several task selections. More specifically, the system generates a large set of paths of the desired length by combining all possible combinations of task selections alternating with all likely student responses to the task. It also estimates the probability of each path's occurrence and the utility of the state at the end of the path. The system then selects the path that leads to a future state with the maximum expected utility, and assigns the first task along that path to the student. The amount of computation required depends strongly on the branching factor, which is the number of possible tasks the tutor can choose and the number of possible student responses. The quality of the task selection depends strongly on having an accurate estimates of the maximum expected utility, which in turn depends on the accuracy of the model of how doing a specific task will affect the student's state. A few tutoring systems have use POMDP algorithms or related algorithms (Cakmak & Lopes, 2012; Clement, Oudeyer, Roy, & Lopes, 2014; Lopes, Clement, Roy, & Oudeyer, 2015; Muldner & Conati, 2007; Murray & VanLehn, 2000, 2006; Murray et al., 2004; Pek & Poh, 2000a, 2000b; Rafferty et al., 2011; Whitehill, 2012). However, the combinatorics tend to limit the use of POMDPs to selection among a small set of tasks.

In lieu of a feasible general solution, a variety of special case solutions have been explored. They all depend strongly on the relationships of the tasks in the candidate sets. Here are some common cases which we are *not* studying.

Suppose the set of candidate tasks contains duplicates, or equivalently, that when a task selected from the set of candidates, it is not removed from the set. This case occurs when students are memorizing vocabulary items, the multiplication table or other facts, and repetition is important for

strengthening memory. As a general rule of thumb, when the same task is given multiple times, students remember its fact longer if there is more time between repetitions (the "spacing effect"). However, this also tends to cause errors, which can slow down or confuse students. Several adaptive task selection algorithms strike a balance between spacing and error (Cepeda, Pashler, Vul, Wixted, & Roher, 2006; de Croock, van Merriënboer, & Paas, 1998; Lindsey, Mozer, Huggins, & Pashler, 2013; Lindsey, Shroyer, Pashler, & Mozer, 2014; Melton, 1970; Pavlik & Anderson, 2008). In our framework, each fact is a knowledge component, and the q-matrix is diagonal. Knowledge components (facts) are assumed to be learned independently of each other.

Another case occurs when the tasks are not identical, but nonetheless all address the same knowledge component(s). That is, from a knowledge component perspective, the tasks are identical. In this case, other factors than knowledge components play a key role in task selection. In particular, many systems try to match the difficulty of the task to the competence of the student. For instance, one of the earliest math tutoring systems classified tasks into five difficulty levels. The adaptive algorithm was simple:

> At the present time we are moving the students up and down the levels of difficulty on the basis of the previous day's performance. If more than 80% of the exercises are correct, the student moves up one level, unless he is already at the top level. If less than 60% of the exercises are correct, the student moves down a level, unless he is already at the bottom. If his percentage of correct answers falls between 60% and 80%, he stays at the same level. (Suppes, 1967, p. 15)

Many other systems have used the same basic idea, and some evaluations have shown benefits of such adaptive task selection methods (Corbalan, Kester, & van Merriënboer, 2008; Salden, Paas, Broers, & Van Merriënboer, 2004; Salden, Paas, Jeroen, & van Merriënboer, 2006; Weber & Brusilovsky, 2001). Some adaptive task selection has been done heuristically using a variety of factors besides difficulty. (Hosseini, Hsiao, Guerra, & Brusilovsky, 2015; McArthur, Stasz, Hotta, Peter, & Burdorf, 1989; Melis et al., 2001; Pek & Poh, 2000a, 2000b; Vizcaino, Contreras, Favela, & Prieto, 2000).

This approach, where all tasks address the same knowledge component(s) and vary only in their difficulty, is so simple that one might wonder why any more complexity is warranted. However, if different tasks actually address different pieces of knowledge, then adaptive task selection should probably not use a single number, as pointed out by Shute, Hansen, and Almond (2008). For instance, suppose in a lesson on quadratic functions, half the tasks use tables and the other half use graphs. When the sequence of tasks is fixed, then the author can arrange the sequence so that it alternates between tables and graphs. The author thus insures that both representations of quadratic functions are adequately exercised. However, with adaptive task selection, the system might just choose all graph tasks or all table tasks. For adaptive task selection, a better solution is to define three knowledge components: quadratic functions in general, graphs of quadratic functions and tables of quadratic functions. Thus, the student's state now consists of three numbers: their competence on each of the three knowledge components. This representation allows the task selection algorithm to select tasks that adequately cover the intended knowledge.

Another case occurs when candidate tasks that do not share knowledge components nonetheless interact beneficially. That is, having the student's do one task helps them learn the knowledge required for another task. This commonly occurs in concept formation or induction. For instance, if a botany student is learning how to distinguish trees by their leaves, then presenting an oak leaf followed immediately by a maple leaf will cause more learning of both than presenting palm leaves in between, because the oak and maple have similar leaves. This is called discrimination learning. There were many studies of how to expedite it, including at least one that used a dynamic, adaptive task selection algorithm (Park & Tennyson, 1980).

Another case occurs when there are pre-requisite relationships among either the candidate tasks or their knowledge components. That is, one knowledge component may be very difficult to learn unless a different knowledge component is mastered first. There are many methods for inferring such relationship from data and using them to select tasks, mostly stemming from Falmagne's seminal work on knowledge spaces (Falmagne, Koppen, Villano, Doignon, & Johannesen, 1990).

The case we are studying assumes that the set of candidate tasks does not have duplicates (and when a task is assigned, it is removed from the set of candidates), that most tasks have different sets of required knowledge components, that there are no pre-requisite relationships among the knowledge components nor the tasks and that discrimination learning is not important. The next section reviews algorithms, knowledge representations and empirical results from prior research on this particular task selection problem.

2.1. Algorithms from prior work

One approach is to select tasks using the same algorithms as used by adaptive testing systems. These systems select a task that will maximize the information gained about the student's competence from the student's answer. Typically, this means that the student has a 50/50 chance of getting the task correct. Shute et al. (2008) showed that learning was not impaired by this assessment method. Thus, if efficient embedded assessment is the primary concern and learning is a secondary concern, this technique should be considered.

When learning is the primary concern, and every task addresses just one knowledge component, then it makes sense to choose a task whose knowledge component is furthest from mastery. That is, for a given student, when all the knowledge components are ranked according to the student's current degree of mastery of them, this knowledge component is minimal. With each learning opportunity, the amount of potential learning decreases (Koedinger, Corbett, & Perfetti, 2012), often following to a power-law or exponential curve (Newell & Rosenbloom, 1981). Thus, the largest learning gains come from selecting the knowledge component with the lowest competence. This means that students are usually assigned the task that is hardest for them among the remaining candidate tasks. If the goal is to optimize a combination of learning, motivation and efficiency, then this choose-the-least-mastered approach may not be viable.

When a single task can address multiple knowledge components, the selection algorithm becomes more complex. For instance, a common approach is to select a task that has the maximum number of unmastered knowledge components (Barr, Beard, & Atkinson, 1976; Koedinger, Pavlik, Stamper, Nixon, & Ritter, 2011). If this approach is followed, then one should use a student modeling algorithm that properly handles assignment of blame, such as DINA, NIDA (Junker & Sijtsma, 2001) or Conjunctive Knowledge Tracing (Koedinger et al., 2011). It also tends to choose the hardest task for the student, which may slow the student down or discourage them.

If a student succeeds on a task, then the system increases its estimates of the student competence on the task's knowledge components. This often means that the next task selected will not involve these knowledge components, but will instead focus on weaker ones. On the other hand, if the student fails on a task, then the system decreases its estimates of student competence on the task's knowledge components, so these knowledge components remain the least mastered ones, and the system's next task tends to be similar to this one. Thus, the system exhibits a lose-stay-win-shift pattern. It keeps giving tasks involving a set of weak knowledge components until the student finally starts getting the tasks right; then it shifts to a new set of knowledge components. This could be discouraging to students.

In short, although several algorithms have focused on maximizing learning by selecting tasks that have the weakest knowledge components, it is not clear whether this kind of adaptive task selection leads to learning or disengagement. Unfortunately, the empirical record is sparse, as the next section indicates.

2.2. Empirical results from prior work

So far, task selection methods have been briefly reviewed, but no empirical results have been presented. An exhaustive review is beyond the scope of this article, but the results listed below should provide an overall impression of the fragmented empirical state of the field.

- Barr et al. (1976) compared their adaptive task selection method (selecting a task with the maximum number of unmastered knowledge components) to a simpler method over ten 1-hour sessions where students used a tutoring system that taught computer programming. They found no difference in post-test scores.
- Park and Tennyson (1980) compare their adaptive task selection method with random selection of tasks. They found a significant difference in post-test scores, with a moderate effect size ($d = 0.6$). However, their system taught students how to discriminate among concepts, and their adaptive task selection used heuristics so that concepts that had been confused by the student appeared in consecutive tasks. This is sometimes called "discrimination drill" in the concept formation and inductive learning literature, which has thoroughly investigated non-adaptive task orderings (e.g. Birnbaum, Kornell, Bjork, & Bjork, 2013; Große & Renkl, 2003).
- Weber and Brusilovsky (2001) showed that novices profited from using an adaptive task selective version of their Lisp tutoring system, whereas students with more incoming knowledge did not.
- Several studies (Corbalan et al., 2008; Salden et al., 2004, 2006) showed that adaptive task selection decreased time-to-mastery in an air traffic control task, but did not affect transfer. In the comparison condition, students received a task sequence generated for a different student.
- Shute et al. (2008) showed that an adaptive task selection algorithm designed for computerized adaptive testing did not harm student's learning compared to a fixed sequence of tasks.

Although there are many adaptive task selection systems in the literature, there are surprisingly few studies of their impact on learning. This project's study is a modest contribution.

2.3. Populating the Q-matrix

When different tasks can address different knowledge components, then the adaptive task selector must know which knowledge components go with which task. This is traditionally called a Q-matrix (Tatsuoka, 1996). It is a two-dimensional matrix, with one dimension listing tasks and the other listing knowledge components. In most implementations, a cell is 1 if correctly solving the cell's tasks requires applying the cell's knowledge component; it is 0 otherwise. As human experts author questions, they typically also tag the question with the knowledge components it requires, and this tagging populates the Q-matrix.

Human authors often overlook knowledge components which are obvious to an expert but not obvious to a novice, so several projects have attempted to infer Q-matrices from data. Some infer a Q-matrix by classifying the text of the task (Karlovcec, Cordova-Sanchez, & Pardos, 2012; Rose et al., 2005). Some infer a Q-matrix by reducing the dimensionality of the response data, viewed as a matrix associating students to tasks (Barnes, Stamper, & Madhyastha, 2006; Desmarais, 2011; Desmarais, Beheshti, & Naceur, 2012). Others take an existing Q-matrix and improve its fit to the data (Cen, Koedinger, & Junker, 2006). Psychometric methods for inferring the Q-matrix from data exist (e.g. Chung, 2014). The psychometric methods, along with some of the others listed above, assume that the data come from students who are not learning as the data are being collected. Others require large amounts of data. This project's results provide a more practical alternative.

2.4. Summary

There are varieties of adaptive task selection research problems. The one we are studying assumes that the same task is never assigned twice, and that discrimination learning, pre-requisites and mastery learning are not issues. We also assume that answering a task can require multiple knowledge components, which makes selecting a task somewhat more complex. We propose an algorithm

that does this kind of adaptive task selection, and show that it affects learning. The algorithm uses a Q-matrix that is filled mechanically rather than by human experts.

3. Description of the tutoring system

From the students' point of view, the tutoring system was extremely simple (see Figure 1). A question appeared on the screen along with a box. When the student typed an answer into the box and clicked the Enter button, the system presented feedback. This continued until the student had worked for 30 minutes.

3.1. Generating questions from a semantic network

All the questions were generated from a knowledge base. The generation process was detailed in our previous paper (Zhang, & VanLehn, 2016), so this section's description is brief.

The knowledge base was a semantic network using standard predicates (Clark, Porter, & Works, 2004). It was developed for answering questions about biology (Baral, Vo, & Liang, 2012). It included part–whole hierarchies of both physical objects (e.g. a mitochondria is a part of a cell) and biological processes (e.g. the Calvin Cycles is a part of photosynthesis). The biological processes included properties indicating where they occurred, their inputs and their results/products. The semantic network included a property inheritance hierarchy (e.g. photosynthesis is a subclass of biological process; the cell membrane is a subclass of membrane), that distinguished between classes and instances of classes. Most properties belong to instances instead of classes, so every class has a generic instance. For instance, the fact that photosynthesis produces oxygen is represented as:

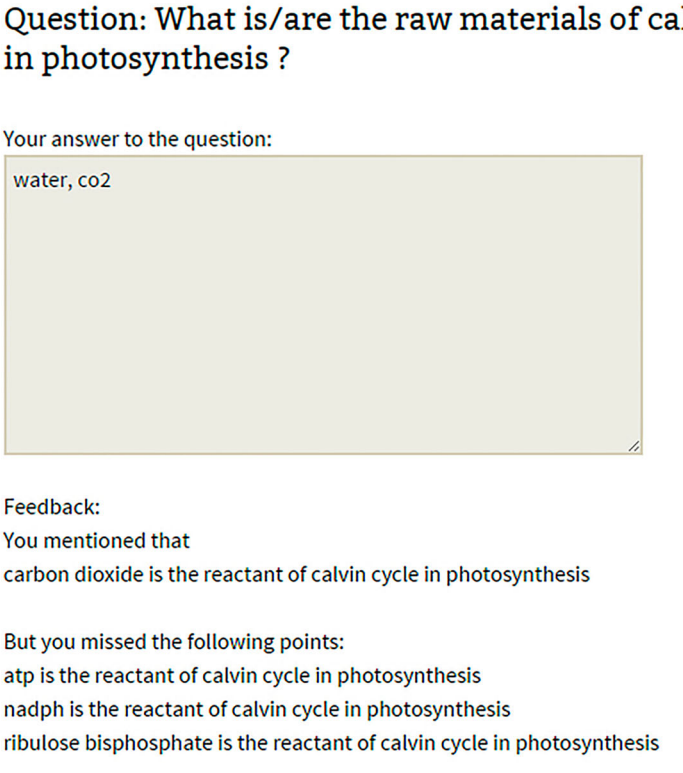

Figure 1. The system feedback on student's answer.

```
has(photosynthesis001, result, oxygen001)
has(photosynthesis001, instance_of, photosynthesis)
has(oxygen001, instance_of, oxygen)
```

In the first line, *result* is a predicate, and *photosynthesis001* and *oxygen001* are generic instances. The second and third lines indicate that the generic instances belong to the classes *photosynthesis* and *oxygen*, respectively.

Questions were generated by matching question schemas against this knowledge base. A question schema consisted of templates and constraints. The templates were English text with variables. The constraints were matched against the knowledge base in order to bind the variables to concepts in the semantic network. Once bound, the templates comprised different questions. As a simple example, suppose the constraints for a question are (in answer-set prolog syntax, where variables are indicated by capitalization):

```
result_of(Process, Result) :-
has(Inst_P, result, Inst_R),
has(Inst_P, instance_of, Process),
has(Inst_R, instance_of, Result).
```

Matching this to the knowledge base would produce multiple bindings for the variables Process and Result. For instance, one binding of the variable Process is to photosynthesis, and when it substituted into the template, "What does $Process produce?", the system generates the question, "What does photosynthesis produce?" Another template in the schema, "What process produces $Result?" generates the questions "What process produces oxygen?"

3.2. Populating the Q-matrix

A Q-matrix indicates which knowledge components are addressed by which tasks (Tatsuoka, 1996). When tasks are authored by humans, then humans must also populate the Q-matrix by hand, which is tedious and prone to errors.

Our Q-matrix was populated automatically. We defined the knowledge components to be the atomic propositions (links) in the knowledge base except those that mention classes. For instance, in the example above, both "What does photosynthesis produce?" and "What process produces oxygen?" were generated by matching

```
Relation1: has(photosynthesis001, result, oxygen001)
Relation2: has(photosynthesis001, instance_of, photosynthesis)
Relation3: has(oxygen001, instance_of, oxygen)
```

Only the first proposition is considered a knowledge component, because the other two propositions function more-or-less as bookkeeping. Thus, the Q-matrix entries for both questions (= tasks) would have just this knowledge component.

3.3. Understanding student answers

Adaptive task selection requires embedded assessment, which entails deciding if the student's response exhibits application of all the knowledge components that it should have, according to the Q-matrix. Because the student's response is in typed natural language, our tutoring system has to understand it, at least partially.

There are a variety of methods for understanding natural language. One is to extract semantic relations from the text and make inference on the relations (Makatchev, Hall, Jordan, Pappuswamy, & VanLehn, 2005). Another is to treat the student's entry as a bag of words, translate it into vector in a vector space developed by principal component analysis or other compression techniques, and classify the vector using algorithms such as Naïve Bayes (Lewis, 1998), decision trees (Apté, Damerau, &

Weiss, 1994) or K-nearest neighbor (Wiemer-Hastings, Wiemer-Hastings, & Graesser, 1999). A third method is to use regular expressions (Bird, 2006). A fourth method is based on keywords (Jackson & Moulinier, 2007).

We used a method based on keywords because it is arguably simpler than all the others. The keywords of a knowledge component were extracted from the relation representing the knowledge component. For example, the keyword for Relation1 in Section 3.2 was "oxygen". To find synonyms for keywords, we used on-line dictionaries including WordNet and Thesaurus. The dictionaries recognized common synonyms like "CO2' and "carbon dioxide", but did not recognize unusual terms like GA3P which is the abbreviation of glyveraldehyde-3-phosphate. There were only a few of these, so they were added manually.

3.4. Checking the accuracy of the natural language understanding

In order to test whether our keyword method was able to achieve satisfactory accuracy, we ran a pilot study comparing decisions made by the machine to those made by a human. This section describes the study.

While the human judge was making decisions, he could see a table that contained four columns: questions, students' answers to the questions, the natural language descriptions of the associated knowledge components to the questions and a column for the judge's Y or N decision. So the human judge was blind to auto-labels. The judge was asked to put "Y" when he thought the knowledge component was mentioned in the answer and "N" when the knowledge component was not mentioned.

The data came from 9 students who answered 48 different questions. In total, there were 153 question–response pairs. Since one question usually covered more than one knowledge component, the judge was asked to make 391 decisions about whether a knowledge component was present or absent from the student's answer.

The confusion matrix (see Table 1) compares the decisions made by the keyword method to the judgments of the human. Most the errors came from false negatives, where the keyword method failed to detect a phrase that was synonymous with a keyword. Here is an example:

Question: What are the 4 stages of cyclic photophosphorylation in light reaction?
Student's answer:
conversion of light energy into chemical energy
absorption of photon
splitting of water
Generating energy carrier

Although the question covered four knowledge components, only one raised a false negative. The knowledge component was "chlorophyll absorbs photons". The keywords of the knowledge component were: chlorophyll, absorb and photon. The human judge decided that the knowledge component was mentioned because of the phrase "absorption of photon". But the system decided that the knowledge component was not mentioned because the word "chlorophyll" was not found in the answer and the nominal form of the keyword "absorb" was not recognized.

Kappa was calculated to measure the agreement between the human judgments and the algorithm's judgments. The kappa was 0.802, which suggests that the keyword method and the human expert agreed with each other at most of time.

Table 1 . Machine (rows) vs. human (columns) judgments.

	Yes	No
Yes	124	5
No	31	230

It is worth mentioning that both the human judge and the keyword method ignored terms that appeared in the answer when they should not have. For example, some students included carbon dioxide in their answer to the question "What does photosynthesis make?" Fortunately, students in the pilot study seldom made these errors. Of the 153 student answers, only 9 contain superfluous terms.

3.5. Embedded assessment

When an expected knowledge component is mentioned in the student's answer to a question, the system's estimate of the student's proficiency in using the knowledge component should be increased. On the other hand, if an expected knowledge component is not included in the answer, its estimated proficiency should be decreased. We used Bayesian Knowledge Tracing (BKT) (Corbett & Anderson, 1995) to represent and update the proficiencies.

In BKT, for each student and each knowledge component, there is a single number between 0 and 1 representing the probability that the student has mastered the knowledge component. This number changes as questions are answered and evidence about the student's proficiency changes.

BKT is typically applied when tasks involve just one knowledge component each and answers are either correct or incorrect; there is no partial credit. A correct answer suggests that the knowledge component may have applied, and an incorrect answer suggests that it may not have applied. However, our tasks often involve multiple knowledge components. Ordinarily, this would require a more elaborate version of BKT that could handle conjunction and perhaps disjunctions of knowledge components (e.g. Koedinger et al., 2011). However, students enter text as answer to such questions, and the keyword method described earlier determines whether the text mentions the expected knowledge components. For instance, if the question is "List the main products of photosynthesis". and the student only enters "oxygen", then one of the knowledge components involved in a completely correct answer would be marked "correct" and the other one ("sugar") would be marked "incorrect". This allows us to use the standard version of BKT.

The BKT method algorithm has three parameters:

- G – the probability of a lucky *guess*. That is, the probability that the student will mention the knowledge component even though the student has not mastered it.
- S – the probability of a *slip*. That is, the probability that the student will fail to mention the knowledge component even though the student has actually mastered it already.
- L – the probability of *learning*. That is, the probability that in the course of answering the question and receiving feedback, the student's competence on this knowledge component will change from unmastered to mastered.

For each student, for each question answered by that student, for each knowledge component required for a completely correct answer, the BKT algorithm performs the following calculation. Let P be the probability, prior to processing this questions' response, that the student had mastered the knowledge component. There are two cases to consider, depending on whether student's answer correctly displayed the knowledge component or incorrectly omitted it. Let P_{correct} be the probability of mastery in the correct case, and $P_{\text{incorrect}}$ be the probability of correctness in the other case. From the definitions of the parameters and Bayes rules, it follows that:

$$P_{\text{correct}} = (1 - S)*P/[(1 - S)*P + G*(1 - P)],$$

$$P_{\text{incorrect}} = S*P/[S*P + (1 - G)*(1 - p)].$$

Next, BKT incorporates a simple assumption about learning, namely that the probability of transitioning from an unmastered to a mastered state is L regardless of what happened during the task. If

we define P_{next} to the probability of mastery of the knowledge component after the processing of this question is complete, then:

$P_{next} = P_{correct} + L*(1 \pm P_{correct})$ if the knowledge component was correct, or
$P_{next} = P_{incorrect} + L*(1 - P_{incorrect})$ otherwise.

In short, BKT has two slightly different formulas for P_{next} depending on whether the answer was correct or incorrect. The left panel of Figure 2 shows how $P_{correct}$ (top 3 curves) and $P_{incorrect}$ (bottom three curves) vary depending on P, S and G. The right panel of Figure 2 shows the two cases of P_{next}, which includes the effects of the learning parameter, L.

3.6. Selecting the next question

Suppose that the student has answered a question, the system has updated the probabilities of mastery of the knowledge components that should have been included in the answer, and it is time to select a new question to pose to the student. The system goes through all the questions in the database that has not yet been asked, calculates their utilities and then selects a question with maximal utility.

The utility is a composite of two factors that correspond to the two stages of the BKT algorithm. That is, we want to maximize P_{next}, the probability of mastery after the student has processed the question, and we also want to maximize $P_{correct} - P$ or $P_{incorrect} - P$, the probability of raising the estimated probability of mastery by getting evidence from this question. The latter is necessary because some questions, namely those with a high probability of guessing and slipping, provide much less evidence than others.

However, because the student has not answered the question, we do not know whether to use $P_{correct}$ or $P_{incorrect}$, so we must do a weighted sum of the two cases. We weight the two cases by their probability of occurrence, namely P and $(1 - P)$, respectively. Thus,

$$\text{Utility} = P*[L*(1 - P_{correct}) + (P_{correct} - P)] + (1 - P)*[L*(1 - P_{incorrect}) + (P_{incorrect} - P)].$$

When the learning rate L is at least 0.3, low probabilities of mastery are preferred. This makes sense, as unlearned knowledge components have a greater chance of becoming learned. But when the learning rate is less than 0.3, then the dominant term in the utility formula is the potential increase due to evidence, that is, $P*(P_{correct} - P) + (1 - P)*(P_{incorrect} - P)$. Thus, there are bumps in Figure 3 around 0.2 and 0.8 because that is where $P_{correct}$ and $P_{incorrect}$ are changing rapidly (see Figure 2). Thus, the curves make some intuitive sense.

When a question is associated with multiple knowledge components, we took the average of the utilities of the knowledge component utilities as the question's utility. This treats all knowledge components as equally important, which seems reasonable in our context. It could be argued that taking the sum of the utilities is more logical. However, that means a question with more knowledge component has more utility than a question with few knowledge components. Moreover, since knowledge components with low probability of mastery are preferred, this would mean preferring questions that required lots of knowledge that was highly unfamiliar. Clearly, this would be a bad policy. Thus, we define question utility as the average rather than the sum of the relevant knowledge component utilities.

We did not have enough data to calibrate BKT, so we chose plausible values for the parameters. We chose $L = 0.3$ because all the knowledge components in the teaching domain appeared to be simple to learn, for example, photosynthesis produces oxygen. G was set as 0.1 because students were required to type answers, which would make guessing unlikely to succeed. S was set as 0.3, which is higher than G, because students were required to type answers and may make both unintentional mistakes and used natural language that our keyword method did not understand properly. For all knowledge components, the initial probability of mastery was set to 0.5.

Figure 2. Probability of mastery calculated by BKT.

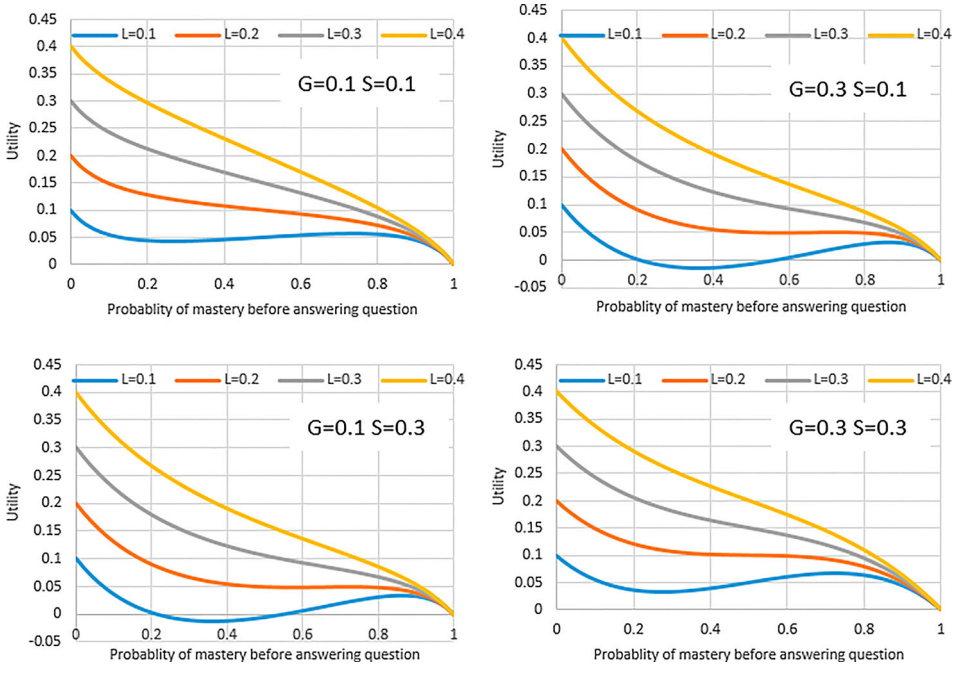

Figure 3. Question's utility vs. probability of mastery.

4. Evaluation

To evaluate our adaptive question selection method, we conducted a between-subject experiment. Both groups of students received questions selected to match their student model. For students in the adaptive group, the system chose a next question that had the *highest* utility for that student. For students in the mal-adaptive group, the system chose a next question that had the *lowest* utility for that student. Given the paucity of empirical results and the frequent appearance of null results, we wanted to test first whether adaptive task selection made any difference in learning at all. Thus, we compare maximization of learning to minimization of learning. If this contrast is reliable, our next experiment would be to compare maximization of learning to a baseline task selection method, such as a fixed sequence of biology questions. However, such results of such an experiment are a function of the baseline task selection method. In particular, not all fixed sequences of questions are equally effective, so there is a risk of producing misleading results by accidentally choosing a particularly ineffective or effective fixed sequence of tasks. Ideally, the baseline condition would use a task sequence that is in widespread use in schools. We are not aware of any such sequence. Thus, we felt it wise to start by seeing if adaptive task selection had any impact at all to learning gains, and later face the methodological challenge of finding or constructing an authentic baseline task selection sequence.

The study was approved by our university's Institutional Review Board. All the participants of the study were college students who were not biology majors. Students who participated in the study earned $10 per hour. There were 42 students in total, 21 students in adaptive group and the 21 subjects in mal-adaptive group.

4.1. Procedure

The experiment was divided into three parts: pre-test, training session where the manipulation took place and the post-test. The pre-test and post-test used the same set of multiple-choice questions, such as the one in Figure 4. The order of the pre-test questions was different from the order of

the post-test questions. There was no time limit for pre-test and post-test. Students spent on average only 5 –10 minutes per test.

During the training session, different students answered different questions, as assigned by the system. There were 48 machine-generated training questions stored in the database, but students answered only as many as they could in the 30 minutes allotted for training. The training session was limited to 30 minutes because we did not want students to run out of questions before the end of training session, and in pilot studies, we found that no one could finish all 48 questions within 30 minutes. Unlike the multiple-choice questions on the pre-test and post-test, students needed to type their answers to the training questions. Participants were allowed to use a researcher-provided biology textbook while answering the questions.

4.2. Results

Students were asked to answer as many questions as they could during the 30-minute training period. Students in both groups answered on average exactly 14.76 questions. This suggests that the set of questions were equivalent in the time to type in the answers plus the time to think of answers. Thus, any difference in pre-to-post gain may be due to the exact nature of the questions chosen.

The experiment aimed to answer two questions:

(1) Would students learn with the machine-generated questions?
(2) Would students learn more with adaptive selection than the mal-adaptive selection?

To answer the first question, we compared the mean of the students' scores on the post-test to the mean of the students' scores on the pre-test. Students in both of the groups increased their test scores significantly. Students in the mal-adaptive group scored 3.32 (SD = 1.57) on the pre-test and 4.74 (SD = 1.24) on the post-test, which was reliable according to a two-tailed *T*-test ($p = .0023$, $T = 3.26$) and comprised a large effect size ($d = 1.01$). Students in the adaptive group scored 2.86 (SD = 1.49) on pre-test and 5.29 (SD = 1.52) on post-test, which was also reliable according to a two-tailed *T*-test ($p = 0.001$, $T = 5.22$) and comprised an even larger effect size ($d = 1.61$).

In order to determine if the adaptive group learned more than the mal-adaptive group, three common measures were used: gain scores, normalized gain scores and ANCOVA.

The first method of comparing learning is to use gain scores, which are calculated by subtracting a student's pre-test score from their post-test score. The average learning gain of mal-adaptive group was 1.42 (SD = 1.78), and the average learning gain of adaptive group was 2.43 (SD = 1.29). This difference is reliable (two-tailed *T*-test, $p = .043$, $T = 2.097$) and a moderately large effect size ($d = 0.65$). So

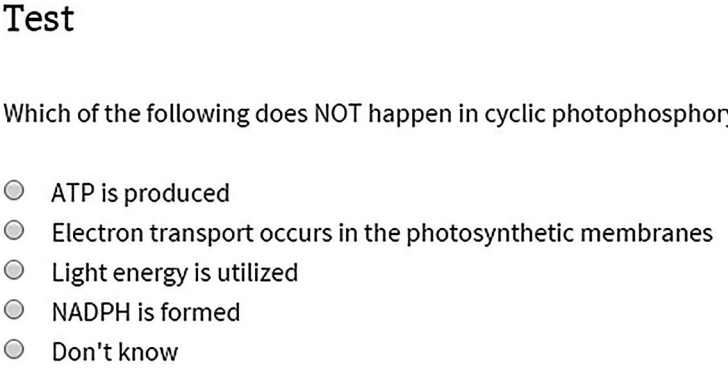

Figure 4. Test question sample.

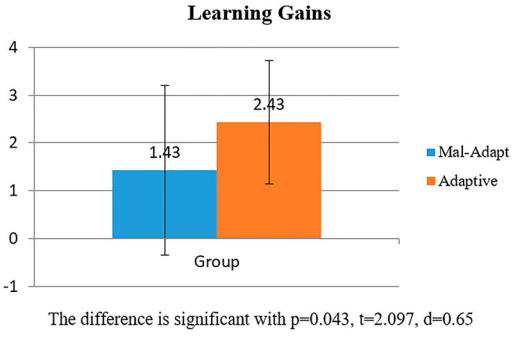

Figure 5. The learning gain of adaptive treatment group vs. the learning gain of mal-adaptive treatment group.

the adaptive question selection did significantly increase students' learning gains compared to mal-adaptive selection of questions (see Figure 5 and Table 2).

Yet another way to compare learning is to use normalized gain scores, which are defined as the post-test score minus the pre-test score divided by the total score minus the pre-test score. The difference between the normalized gain scores of the two groups was reliable (two-tailed *T*-test, $p = .024$, $T = 2.357$) and the effect size was moderately larger ($d = 0.72$).

Instead of comparing gain scores between the two groups, an alternative way is to compare students' post-test scores with their pre-test scores as covariates in an ANCOVA. Applying an ANCOVA requires that the pretest score is linearly correlated with the posttest score (Eric, 1998), which it was in this case ($p = .0089$). The ANCOVA reported the difference between groups' adjusted post-test scores was marginally reliable ($p = .075$, $F = 3.348$) and comprised a moderately large effect size ($d = 0.58$) (see Figure 6, left).

In order to resolve the discrepancy between the different methods for comparing learning, we looked for and found an aptitude–treatment interaction. The difference between gain scores of the two groups is an unbiased estimation of the treatment's effect when gain score is not affected by pretest score. To test this assumption, we calculated Pearson correlation coefficient between the pretest score and the gain score in our experiment. The gain score was moderately correlated to the pretest score ($r = -.61$). The normalized gain score was somewhat less correlated with the pretest score ($r = -.40$). This suggests that students with lower pre-test scores are less affected by the manipulation than students with higher pre-test scores.

A typical method for examining such aptitude–treatment interactions is to split the sample into high-pretest and low-pretest groups using the median score. Students whose scores were equal to or above the median (median = 3) were classified into the high pre-test group ($N = 28$), and the rest were classified into the low pre-test group ($N = 14$). According to ANCOVA, the adaption made a significant difference in the high pre-test group ($p = .027$, $F = 5.519$, $d = 0.89$), see Figure 6 right, but not for the low pre-test group ($p = .828$, $F = 0.0496$, $d = 0.0687$). In order to determine if this result was sensitive to the cut point, we ran ANCOVA tests splitting the sample at 2 and at 4. Both

Table 2. Comparison between experimental group and control group.

	Pre-test	Post-test	Learning gain
Mal-adaptive	3.33 (SD = 1.56)	4.76 (SD = 1.22)	1.43(1.78), $p = .0023$, $d = 1.01$
Adaptive	2.86 (SD = 1.49)	5.29 (SD = 1.52)	2.43(1.29), $p<0.001$, $d = 1.61$

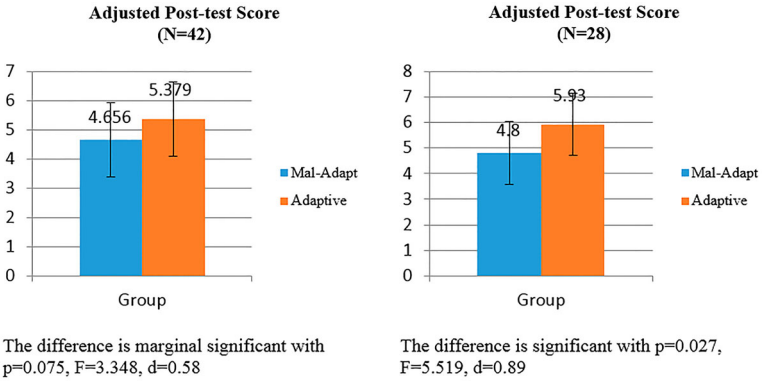

Figure 6. The adjusted post-test score of adaptive treatment group vs. that of mal-adaptive treatment group.

tests lead to significant differences ($p = .034$ and $p = .038$, respectively) for the high pre-test group but not for the low pre-test group.

4.3. Discussion of the evaluation

Our results suggest that for students with higher pre-test scores, adaptive question selection fosters more learning than mal-adaptive question selection. The difference between the treatments was only in whether they maximized or minimized the utility of the selected question for the individual student, and utility was defined in terms of expected learning gains. The embedded assessment and the utility calculations assume that all students were trying equally hard, so their performance was equally a function of their competence. This may not have been the case. Suppose there were some low-diligence students in the sample who exerted little effort on the tests and the training, and mostly just guessed at answers. They would tend to get low pre-test scores, and the estimates of their competence would be inaccurate. Thus, the system's choice of questions for the low-diligence students would be essentially random in both the adaptive and mal-adaptive conditions. Thus, the low-diligence students' learning gains would be about the same in both conditions. This would explain the observed pattern in the results.

5. Conclusion

Adaptively task selection has been explored since at least the 1960s. It often requires considerable judgment from humans on which knowledge components go with which tasks and other factors. However, we managed to provide an automatic way to build adaptation into a question-asking system by utilizing a semantic network and typed answer understanding. Except for providing a few synonyms for esoteric biology terms, no human knowledge was inserted into the question generation, response interpretation and adaptive question selection processes.

However, just because machine-generated adaptation is feasible does not mean it will achieve its purpose, which is to increase learning gains. A between-subjects experiment was conducted to evaluate the effectiveness of the adaptive task selection algorithm. Students with adaptive question selection had larger learning gains than those with mal-adaptive question selection, and both of the students scored higher on the post-test than the pre-test. The benefits were particularly pronounced for students with high pre-test scores, perhaps because some of the low pre-test students were not taking the experiment seriously, so their behavior was more random than knowledge-based.

Although the system achieved satisfying results, it had many aspects that could be improved. Perhaps the most important one is that the system was only able to detect the presence or

absence of the correct knowledge components in student's answer. It failed to detect the presence of unexpected or incorrect knowledge components. To solve this problem would require collecting common incorrect responses for each question. These could be added to the set of expected keywords, so that the system can recognize them in student's answer. Alternatively, the system could use multiple-choice questions instead of open-response questions, and include the expected incorrect response among the choices (foils). However, suppose the system detects that a student has an incorrect belief, such as "carbon dioxide is a product of photosynthesis". Does that mean it should decrement the probability of mastery of the knowledge components that correctly define the products of photosynthesis? This would be part of our future work.

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

This research was supported by The Diane and Gary Tooker Chair for Effective Education in Science, Technology, Engineering and Math, and by the National Science Foundation under grant DUE-1525197.

Notes on contributors

Lishan Zhang is a Research Assistant in the School of Computing, Informatics and Decision Science Engineering at Arizona State University. He received a Ph. D. from in Computer Science from Arizona State University in 2015. He has published 2 journal articles and 6 conference papers. His main research area is intelligent tutoring system.

Kurt VanLehn is the Tooker Professor of STEM Education in the School of Computing, Informatics and Decision Science Engineering at Arizona State University. He received a PhD from MIT in 1983 in Computer Science, and worked at BBN, Xerox PARC, CMU and the University of Pittsburgh. He founded and co-directed two large NSF research centers (Circle; the Pittsburgh Science of Learning Center). He has published over 125 peer-reviewed publications, is a fellow in the Cognitive Science Society, and is on the editorial boards of Cognition and Instruction and the International Journal of Artificial Intelligence in Education. Dr VanLehn has been working in the field of intelligent tutoring systems since such systems were first invented. Most of his current work seeks new applications of this well-established technology. For example, three current projects are: (1) FACT, a tablet-based classroom system for helping teachersmore deeply analyze student work during complex math formative assessments; (2) Dragoon, an intelligent tutoring system that imparts skill in constructing models of dynamic systems so rapidly that it can be used in science classes to help students understand the systems more deeply; and (3) OPE, an intelligent tutoring system for organic chemistry aimed at keeping students motivated and improving their self-regulated learning skills.

References

Apté, C., Damerau, F., & Weiss, S. M. (1994). Automated learning of decision rules for text categorization. *ACM Transactions on Information Systems (TOIS)*, *12*(3), 233–251.

Arroyo, I., Mehranian, H., & Woolf, B. P. (2010). *Effort-based tutoring: An empirical approach to intelligent tutoring*. Paper presented at the 3rd International Conference on Educational Data Mining.

Baral, C., Vo, N. H., & Liang, S. (2012). *Answering Why and How questions with respect to a frame-based knowledge base: a preliminary report*. Paper presented at the ICLP (Technical Communications).

Barnes, T., Stamper, J., & Madhyastha, T. (2006). *Comparative analyses of concept derivation using the q-matrix method and facets*. Paper presented at the Educational Data Mining Workshop at the AAAI 21st National Conference on Artificial Intelligence (AAAI2006), Boston, MA.

Barr, A., Beard, M., & Atkinson, R. C. (1976). The computer as a tutorial laboratory: The Stanford BIP project. *International Journal of Man-Machine Studies*, *8*, 567–596.

Beck, J., Woolf, B. P., & Beal, C. (2000). ADVISOR: A machine learning architecture for intelligent tutor construction. In *Proceedings of the seventeenth national conference on artificial intelligence* (pp. 552–557). Menlo Park, CA: AAAI Press.

Bird, S. (2006). *NLTK: The natural language toolkit*. Paper presented at the Proceedings of the COLING/ACL on Interactive presentation sessions.

Birnbaum, M. S., Kornell, N., Bjork, E. L., & Bjork, R. A. (2013). Why interleaving enhances inductive learning: The roles of discrimination and retrieval. *Memory & Cognition*, *41*, 392–402.

Bloom, B. S. (1974). Time and learning. *American Psychologist*, *29*(9), 682–688.
Cakmak, M., & Lopes, M. (2012). *Algorithmic and human teaching of sequential decision tasks*. Paper presented at the AAAI Conference on Artificial Intelligence, Toronto, Canada.
Cen, H., Koedinger, K. R., & Junker, B. (2006). Learning factors analysis – a general method for cognitive model evaluation and improvement. In M. Ikeda, K. Ashley & T.-W. Chan (Eds.), *Intelligent tutoring systems: 8th International Conference, ITS 2006* (pp. 164–175). Berlin: Springer.
Cepeda, N. J., Pashler, H., Vul, E., Wixted, J. T., & Roher, D. (2006). Distributed practice in verbal recall tasks: A review and quantitative synthesis. *Psychological Bulletin*, *132*(3), 354–380.
Chung, M.-T. (2014). *Estimating the Q-matrix for cognitive diagnosis models in a Bayesian framework* (Ph.D.). Columbia University.
Clark, P., Porter, B., & Works, B. P. (2004). *KM—The knowledge machine 2.0: Users manual*. Department of Computer Science, University of Texas at Austin.
Clement, B., Oudeyer, P.-Y., Roy, D., & Lopes, M. (2014). *Online optimization of teaching sequences with multi-armed bandits*. Paper presented at the Educational Data Mining.
Corbalan, G., Kester, L., & van Merriënboer, J. J. G. (2008). Selecting learning tasks: Effects of adaptation and shared control on learning efficiency and task involvement. *Contemporary Educational Psychology*, *33*, 733–756.
Corbett, A., & Anderson, J. R. (1995). Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction*, *4*, 253–278.
de Croock, M. B. M., van Merriënboer, J. J. G., & Paas, F. G. W. C. (1998). High versus low contextual interference in simulation-based training of troubleshooting skills: Effects on transfer performance and invested mental effort. *Computer in Human Behavior*, *14*(2), 249–267.
Desmarais, M. C. (2011). *Conditions for effectively deriving a q-matrix from data with non-negative matrix factorization*. Paper presented at the Educational Data Mining.
Desmarais, M. C., Beheshti, B., & Naceur, R. (2012). *Item to skills mapping: Deriving a conjunctive Q-matrix from data*. Paper presented at the Intelligent Tutoring Systems.
Eric, M. (1998). Covariance adjustment versus gain scores–revisited. *Psychological Methods*, *3*, 309–327. doi:10.1037//1082–989X.3.3.309
Falmagne, J., Koppen, M., Villano, M., Doignon, J., & Johannesen, L. (1990). Introduction to knowledge spaces: How to build, test and search them. *Psychological Review*, *97*, 201–224.
Große, C. S., & Renkl, A. (2003). Example-based learning with multiple solution methods fosters understanding. In F. Schmalhofer & R. Young (Eds.), *Proceeding of the European Conference of the Cognitive Science Society* (pp. 163–168). Mahwah, NJ: Erlbaum.
Guskey, T. R., & Gates, S. L. (1986). Synthesis of research on the effects of mastery learning in elementary and secondary classrooms. *Educational Leadership*, *43*(8), 73–80.
Hosseini, R., Hsiao, I.-H., Guerra, J., & Brusilovsky, P. (2015). *Off the beaten path: The impact of adaptive content sequencing on student navigation in an open student modeling interface*. Paper presented at the Artificial Intelligence in Education.
Jackson, P., & Moulinier, I. (2007). *Natural language processing for online applications: Text retrieval, extraction and categorization (Vol. 5)*. Amsterdam: John Benjamins Publishing.
Junker, B., & Sijtsma, K. (2001). Cognitive assessment models with few assumptions, and connections with nonparametric item response theory. *Applied Psychological Measurement*, *25*(3), 258–272.
Karlovcec, M., Cordova-Sanchez, M., & Pardos, Z. A. (2012). *Knowledge component suggestion for untagged content in an intelligent tutoring system*. Paper presented at the Intelligent Tutoring Systems.
Koedinger, K. R., Corbett, A., & Perfetti, C. (2012). The knowledge-learning-instruction (KLI) framework: Bridging the science-practice chasm to enhance robust student learning. *Cognitive Science*, *36*(5), 757–798.
Koedinger, K. R., Pavlik, P. I., Stamper, J., Nixon, T., & Ritter, S. (2011). *Avoiding problem selection thrashing with conjunctive knowledge tracing*. Paper presented at the International Conference on Educational Data Mining, Eindhoven, NL.
Kulik, C., Kulik, J., & Bangert-Drowns, R. (1990). Effectiveness of mastery learning programs: A meta-analysis. *Review of Educational Research*, *60*(2), 265–306.
Lewis, D. D. (1998). Naive (Bayes) at forty: The independence assumption in information retrieval. *Machine learning: ECML-98*, *1398*, 4–15. Springer.
Lindsey, R. V., Mozer, M. C., Huggins, W. J., & Pashler, H. (2013). *Optimizing instructional policies*. Paper presented at the Advances in Neural Information Processing Systems 26 (NIPS 2013).
Lindsey, R. V., Shroyer, J. D., Pashler, H., & Mozer, M. C. (2014). Improving students long-term retention through personalized review. *Psychological Science*, *25*(3), 639–647.
Lopes, M., Clement, B., Roy, D., & Oudeyer, P.-Y. (2015). Multi-armed bandits for intelligent tutoring systems. *Journal of Educational Data Mining*, *7* (2). Advance online publication. *arXiv preprint arXiv:1310.3174*.
Makatchev, M., Hall, B. S., Jordan, P. W., Pappuswamy, U., & VanLehn, K. (2005). *Mixed language processing in the Why2-Atlas tutoring system*. Paper presented at the Proceedings of the Workshop on Mixed Language Explanations in Learning Environments, AIED2005.
McArthur, D., Stasz, C., Hotta, J., Peter, O., & Burdorf, C. (1989). *Skill-oriented task sequencing in an intelligent tutor for basic algebra*. Santa Monica, CA: The RAND Corporation.

Melis, E., Andres, E., Budenbender, J., Frischauf, A., Goguadze, G., Libbrecht, P., … Ullrich, C. (2001). ActiveMath: A generic and adaptive web-based learning environment. *International Journal of Artificial Intelligence in Education*, *12*, 385–407.

Melton, A. W. (1970). The situation with respect at the spacing of repetitions and memory. *Journal of Verbal Learning and Verbal Behavior*, *9*(5), 596–606.

Muldner, K., & Conati, C. (2007). Evaluating a decision-theoretic approach to tailored example selection. In *Proceedings of IJCAI 2007, the 20th International Joint Conference in Artificial Intelligence* (pp. 483–489). Menlo Park, CA: AAAI Press.

Muñoz, K., Mc Kevitt, P., Lunney, T., Noguez, J., & Neri, L. (2010). PlayPhysics: An emotional games learning environment for teaching physics. In *Knowledge science, engineering and management* (pp. 400–411). Berlin: Springer.

Murray, R. C., & VanLehn, K. (2000). DT Tutor: A decision-theoretic, dynamic approach for optimal selection of tutorial actions. In G. Gauthier, C. Frasson, & K. VanLehn (Eds.), *Intelligent Tutoring Systems: 5th International Conference, ITS2000* (pp. 153–162). Berlin: Springer.

Murray, R. C., & VanLehn, K. (2006). A comparison of decision-theoretic, fixed-policy and random tutorial action selection. In M. Ikeda, K. Ashley, & T.-W. Chan (Eds.), *Intelligent Tutoring Systems: 8th International Conference ITS 2006* (pp. 114–123). Berlin: Springer.

Murray, R. C., VanLehn, K., & Mostow, J. (2004). Looking ahead to select tutorial actions: A decision-theoretic approach. *International Journal of Artificial Intelligence and Education*, *14*(3–4), 235–278.

Newell, A., & Rosenbloom, P. S. (1981). Mechanisms of skill acquisition and the law of practice. In J. R. Anderson (Ed.), *Cognitive skills and their acquisition* (pp. 1–56). Hillsdale, NJ: Lawrence Erlbaum Associates.

Park, O.-C., & Tennyson, R. D. (1980). Adaptive design strategies for selecting number and presentation order of examples in coordinate concept acquisition. *Journal of Educational Psychology*, *72*(3), 362–370.

Pavlik, P. I., & Anderson, J. R. (2008). Using a model to compute the optimal schedule of practice. *Journal of Experimental Psychology: Applied*, *14*(2), 101–117.

Pek, P.-K., & Poh, K.-L. (2000a). Framework of a decision-theoretic tutoring system for learning of mechanics. *Journal of ScienceEducation and Technology*, *9*(4), 343–356.

Pek, P.-K., & Poh, K.-L. (2000b). Using decision networks for adaptive tutoring. In S. S. Young, J. Greer, H. Maurer, & Y. S. Chee (Eds.), *Proceedings for the International Conference on Computers in Education/International Conference on Computer-Assisted Instruction* (pp. 1076–1084). Taiwan: AACE/APC-National Tsing Hua University.

Rafferty, A. N., Brunskill, E., Griffiths, T. L., & Shafto, P. (2011). Faster teaching by POMDP planning. In G. Biswas (Ed.), *Artificial intelligence in education* (pp. 280–287). Berlin: Springer-Verlag.

Rose, C. P., Donmez, P., Gweon, G., Knight, A., Junker, B., & Heffernan, N. (2005). *Automatic and semi-automatic skill coding with a view towards supporting on-line assessment*. Paper presented at the Artificial Intelligent in Education.

Salden, R. J. C. M., Paas, F. G. W. C., Broers, N. J., & Van Merriënboer, J. J. G. (2004). Mental effort and performance as determinants for the dynamic selection of learning tasks in air traffic control training. *Instructional Science*, *32*, 153–172.

Salden, R. J. C. M., Paas, F. G. W. C., Jeroen, J. G., & van Merriënboer, J. J. G. (2006). Personalize adaptive task selection in air traffic control: Effects on training efficiency and transfer. *Learning and Instruction*, *16*, 350–362.

Shute, V. J., Hansen, E. G., & Almond, R. G. (2008). You can't fatten a hog by weighing it–or can you? Evaluating an assessment for learning system called ACED. *International Journal of Artificial Intelligence and Education*, *18*, 289–316.

Slavin, R. E. (1990). Mastery learning re-reconsidered. *Review of Educational Research*, *60*(2), 300–302.

Suppes, P. (1967). On using computers to individualize instruction. In D. D. Bushnell & D. W. Allen (Eds.), *The computer in American education* (pp. 11–24). New York: Wiley.

Tatsuoka, K. (1996). Architecture of knowledge structures and cognitive diagnosis: A statistical pattern recognition and classification approach. In P. Nichols, P. Chipman, & R. Brennan (Eds.), *Cognitively diagnostic assessment* (pp. 327–359). Mahwah, NJ: Erlbaum.

Vizcaino, A., Contreras, J., Favela, J., & Prieto, M. (2000). *An adaptive, collaborative environment to develop good habits in programming*. Paper presented at the Intelligent Tutoring Systems, Berlin.

Weber, G., & Brusilovsky, P. (2001). ELM-ART: An adaptive versatile system for web-based instruction. *International Journal of Artificial Intelligence in Education*, *12*, 351–384.

Whitehill, J. (2012). *A stochastic optimal control perspective on affect-sensitive teaching* (Ph. D.). University of California, San Diego.

Wiemer-Hastings, P., Wiemer-Hastings, K., & Graesser, A. (1999). Improving an intelligent tutor's comprehension of students with Latent Semantic Analysis. Paper presented at the Artificial Intelligence in Education.

Wood, D. J. (2001). Scaffolding, contingent tutoring and computer-supported learning. *International Journal of Artificial Intelligence and Education*, *12*, 280–292.

Zhang, L., & VanLehn, K. (2016). How do machine-generated questions compare to human-generated questions? *Research and Practice in Technology Enhanced Learning*, *11*(1), 1.