Using Knowledge Component Modeling to Increase Domain Understanding in a Digital Learning Game

Huy Nguyen
Carnegie Mellon University
hn1@andrew.cmu.edu

Yeyu Wang
University of Pennsylvania
yeyuw@upenn.edu

John Stamper
Carnegie Mellon University
jstamper@cs.cmu.edu

Bruce M. McLaren
Carnegie Mellon University
bmclaren@cs.cmu.edu

ABSTRACT

Knowledge components (KCs) define the underlying skill model of intelligent educational software, and they are critical to understanding and improving the efficacy of learning technology. In this research, we show how learning curve analysis is used to fit a KC model - one that was created after use of the learning technology - which can then be improved by human-centered data science methods. We analyzed data from 417 middle-school students who used a digital learning game to learn decimal numbers and decimal operations. Our initial results showed that problem types (e.g., ordering decimals, adding decimals) capture students' performance better than underlying decimal misconceptions (e.g., longer decimals are larger). Through a process of KC model refinement and domain knowledge interpretation, we were able to identify the difficulties that students faced in learning decimals. Based on this result, we present an instructional redesign proposal for our digital learning game and outline a framework for post-hoc KC modeling in a tutoring system. More generally, the method we used in this work can help guide changes to the type, content and order of problems in educational software.

Keywords

KC Model, Decimal Number, Digital Learning Game

1. INTRODUCTION

In the view of KC modeling, student's knowledge can be treated as a set of inter-related KCs, where each KC is "an acquired unit of cognitive function or structure that can be inferred from performance on a set of related tasks" [22]. A KC-based student model (which we refer to as *KC model*) has been employed in a wide range of learning tasks, such as supporting individualized problem selection [11], choosing examples for analogical comparison [35] and transitioning from worked examples to problem solving [43]. A good KC model is vital to intelligent educational software, particularly in the design of adaptive feedback, assessment of student knowledge and prediction of learning outcomes [24].

A new area in educational technology that could potentially benefit from KC models is digital learning game. While there has been much enthusiasm about the potential of digital games to engage students and enhance learning, few rigorous studies have demonstrated their benefits over more traditional instructional approaches [32, 34]. One possible reason is that most digital learning games have been designed in a one-size-fits-all approach rather than with personalized instruction in mind [9]. Adopting KC modeling techniques could therefore be an important first step in meeting individual students' learning needs and making digital learning games a more effective form of instruction. A critical question in this direction is whether a KC model can be created after the use of the learning technology, in order to better understand the targeted learning domain and to help in improving the technology.

In our study, we explore this question in the context of a game that teaches decimal numbers and decimal operations to middle-school students. We started with an initial KC model based on problem type (e.g., adding decimals, completing sequences of decimals), then used the human-machine discovery method [51] to derive new KCs and formulate the best fitting model. From this improved model, we first discuss findings about students' learning of decimal numbers and propose potential changes to the instructional materials that address a wider range of learning difficulties - a process known as "closing the loop" [24]. Then, we outline a general framework for adding KC models to educational software in a post-hoc manner and discuss its broader implications in digital learning games.

2. BACKGROUND

In this section, we first present background information about two aspects of student modeling that are relevant to our work: (1) KC modeling, a technique that represents students' knowledge as latent variables, and (2) the current state of student modeling in digital learning games. Then,

Huy Nguyen, Yeyu Wang, John Stamper and Bruce McLaren "Using Knowledge Component Modeling to Increase Domain Understanding in a Digital Learning Game" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 139 - 148

we describe the game environment used for data analysis.

2.1 KC Modeling

Traditionally, KC models have been developed by domain experts, using Cognitive Task Analysis methods such as structured interviews, think aloud protocols and rational analysis [45]. These methods result in better instructional design but are also highly subjective and require substantial human effort. To address this shortcoming, a wide range of prior research has focused on creating KC models through data-driven techniques. Some of the earliest work on identifying and improving KC models was done by Corbett and Anderson [11] with the early LISP tutors. In this work, plotting of learning curves showed "blips" or "peaks" in the curves which indicated new KCs that were not accounted for in the initial model. By using a computational model to fit the data in learning curves, [5] showed how Learning Factor Analysis (LFA) could automate the process of identifying additional KCs in educational software. LFA takes as input a space of hypothesized KCs, which can be discovered through visualization and analysis tools [51]. Once there are several human-generated KC models, they can be combined by merging and splitting skills using machine learning techniques that aim to improve the overall fit [23].

It is important to define a good model, but it is not always clear how to do so. Goodness of fit is best measured by cross validation, but this technique is time consuming and computationally expensive for large datasets. Furthermore, there is no consensus on how cross validation should be performed on educational data [50]. Two related and easy-to-compute metrics are the Akaike information criterion (AIC) and Bayesian information criterion (BIC), which address overfit by measuring prediction accuracy while penalizing complexity. In general, a lower AIC/BIC/cross validation score indicates a better model. In case they do not agree, [50] showed that AIC correlates with cross validation better than BIC, through an analysis of 1,943 KC models in DataShop. However, these scores alone do not portray the full picture; as pointed out by [3], many student modeling techniques that aim to predict student learning achieve negligible accuracy gains, "with differences in the thousandths place," suggesting that they are already close to ceiling performance. In response, [28] brought attention to another important criterion - whether the model is *interpretable* and *actionable*. As the authors argued, even slight improvement can be meaningful if it reveals insights on student learning that generalize to a new context and lead to better, empirically validated instructional designs. For instance, some research has been successful in redesigning tutor units to help students reach mastery more efficiently, based on analysis of previous KC models [24, 27].

Our analysis follows the established process outlined above, in which we started with a basic human-generated KC model, then identified potential improvements using learning curve analysis, and evaluated the new model by AIC, BIC and cross validation. We also derived instructional insights from this model as the first step in closing the loop.

2.2 Student Modeling in Games

As pointed out by [2], knowledge in digital learning games is harder to represent than knowledge in tutoring systems because the students' thinking process, as well as learning objectives, may not be as explicit. The popular student modeling techniques for learning games are those that can represent uncertainty, such as Bayesian Networks (BN) [31] and Dynamic Bayesian Networks (DBN) [8]. For instance, in *Use Your Brainz*, by applying BN to each level of the game to estimate the problem-solving skills of learners, researchers were able to validate their measures of stealth assessment [46]. [10] applied DBN in *Prime Climb*, a math game for learning factorization, to build an intelligent pedagogical agent that results in more learning gains for students. Follow-up work by [30] refined and evaluated the existing DBN, yielding substantial improvement in the model's test performance prediction accuracy, which in turn helps better estimate students' learning states in future studies. As another example, [42] employed DBN to predict responses on post-test questions in *Crystal Island*, an immersive narrative-based environment for learning microbiology.

Recent research has proposed entirely data-driven methods for discovering KC models in a tutoring system [17, 26]. However, most KC models employed in digital learning games have been generated manually by domain experts. For instance, in *Zombie Division*, the KCs were identified by math teachers as common prime factors such as "divide by two" and "divide by three" [2]. Similarly, the designers of *Crystal Island* labeled the general categories of knowledge involved in problem-solving as *narrative*, *strategic*, *scenario solution* and *content* knowledge [42]. The first attempt to refine a human-generated baseline KC model using data-driven techniques in digital learning games was done by Harpstead and Aleven [18]. Their approach, which was applied to *Beanstalk*, a game that teaches the concept of physical balance, is based on [51]'s human-machine discovery method, which is very similar to ours; however, there are notable differences in the learning environments. In particular, the domain of decimal numbers involves many more rules and operations than *Beanstalk*'s domain of beam balancing; in turn, our digital learning game also incorporates more activities (e.g., placing numbers on a number line, completing sequences, assigning numbers to less-than and greater-than buckets). Therefore, our KC modeling process takes into account not just the instructional materials but also elements of the interface and problem types, which could be more generalizable to other learning environments.

2.3 A Digital Learning Game for Decimals

Decimal Point is a single-player game that helps middle-school students learn about decimal numbers and their operations (e.g., adding, ordering, comparing). The game is based on an amusement park metaphor (Figure 1), where students travel to various areas of the park, each with a different theme (e.g., *Haunted House*, *Sports World*), and play a variety of mini-games within each theme area, each targeting a common decimal misconception: `Megz` (longer decimals are larger), `Segz` (shorter decimals are larger), `Pegz` (the two sides of a decimal number are separate and independent) and `Negz` (decimals smaller than 1 are treated as negative numbers) [21, 47]. Each mini-game also involves one of the following *problem types*:

1. `NumberLine` - locate the position of a given decimal number on the number line.

2. `Addition` - add two decimal numbers by entering the carry digits and the result digits.
3. `Sequence` - fill in the next two numbers of a given sequence of decimal numbers.
4. `Bucket` - compare given decimal numbers to a threshold number and place each decimal in a “less than” or “greater than” bucket.
5. `Sorting` - sort a given list of decimal numbers in ascending or descending order.

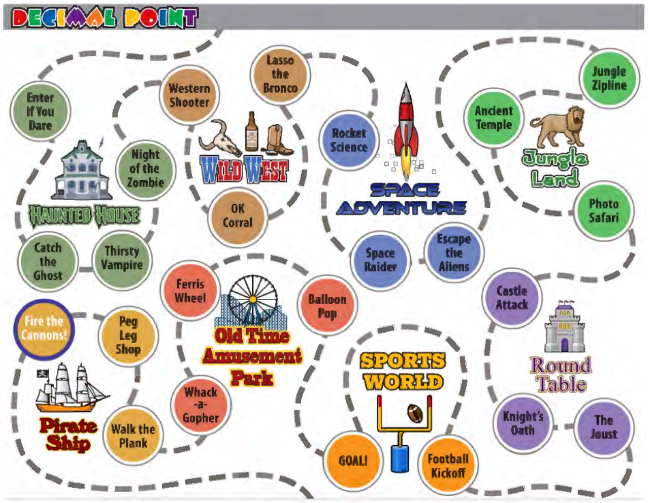
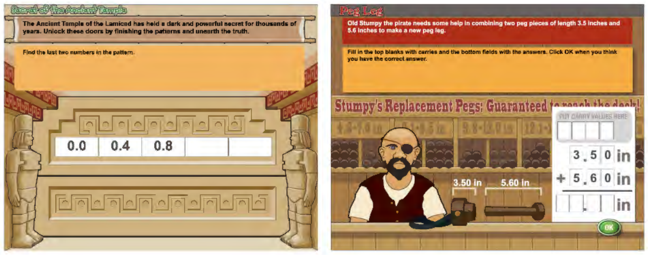


Figure 1: A screenshot of the main map screen.

In each theme area, and across the different theme areas, the problem types are interleaved to improve mathematics learning [41] and introduce variety and interest in gameplay. Figure 2 shows the screenshots of two mini-games - *Ancient Temple* (a `Sequence` game) and *Peg Leg Shop* (an `Addition` game). Each mini-game requires students to solve up to three problems of the same type (e.g., place three numbers on a number line, or complete three number sequences). Students must answer correctly to move to the next mini-game; they also receive immediate feedback about their answers. To further support learning, after a problem has been solved, students are prompted to self-explain their answer by selecting from a multiple-choice list of possible explanations [7].

A prior study by [34] showed that *Decimal Point* promoted more learning and enjoyment than a conventional instructional system with identical decimal content. Follow-up studies by [37] and [19] then tested the effect of student agency, where students can choose the order and number of mini-games they play. These studies revealed no differences in learning or enjoyment between low- and high-agency conditions, but [19] found that students in a high-agency condition had the same learning gains while playing fewer mini-games than those in low-agency, suggesting that the high-agency version led to more learning efficiency.

Post-hoc analyses by [52] examined the different mini-game sequences played by high-agency students and found that, consistent with the reports in [19], those who stopped early learned as much as those who played all mini-games. This result leads to important questions about the right amount of instructional content to maximize learning efficiency. To answer these questions, we would need a more fine-grained measure of student learning using in-game data rather than external test scores. The KC modeling work presented here represents the first step in this direction.



The Ancient Temple of the Lamicad has held a dark and powerful secret for thousands of years. Unlock these doors by finishing the patterns and unearth the truth.

Find the last two numbers in the pattern.

0.0	0.4	0.8		

Old Stumpy the pirate needs some help in combining two peg pieces of length 3.5 inches and 5.6 inches to make a new peg leg.

Fill in the top blanks with carries and the bottom fields with the answers. Click OK when you think you have the correct answer.

(a) Ancient Temple (b) Peg Leg Shop

Figure 2: Screenshots of two mini-games.

2.3.1 *Participants and Design*

We obtained data from two prior studies of *Decimal Point* involving 484 students in 5th and 6th grade, in all study conditions [19, 37], and removed those students who did not finish all of the required materials, reducing the sample to 417 students (200 males, 216 females, 1 declined to respond). The students played either some or all of the 24 mini-games in Figure 1, depending on their assigned agency condition, as described previously. When selecting a mini-game, students would play two instances of that game, with the same interface and game mechanics but different questions. Students in the high-agency condition also had the choice to play a third instance of each mini-game once. In subsequent analyses, we use an index of 1, 2 and 3 to denote the instance number, e.g., *Ancient Temple 1, Ancient Temple 2* and *Ancient Temple 3.* For a detailed description of the experimental design of prior studies, refer to [19, 37].

2.4 Dataset

We analyzed students’ in-game performance data, which was archived in the DataShop repository [49] in dataset number 2906. The dataset covers a total of 613,055 individual transactions, which represent actions taken in the mini-games by 417 students in solving decimal problems.

3. METHODS & RESULTS

We started with the baseline KC models derived from two sets of features that *Decimal Point* was built upon. These initial models were fit using the Additive Factors Model (AFM) method [6], and the learning curves were visualized in DataShop. AFM is a specific instance of logistic regression, with student-correctness (0 or 1) as the dependent variable and with independent variable terms for each student, each KC, and the KC by opportunity interaction. It is a generalization of the log-linear test model [54] produced by adding the KC by opportunity terms. We then chose the model with better fit and analyzed its learning curves. Each model was run on 42,637 observations tagged with KCs.

3.1 Baseline Models

Our first baseline model, called `DecimalMisc`, consists of four KCs that are the misconceptions targeted by the mini-games: `Megz, Segz, Negz, Pegz` [21]. Because each mini-game was designed based on a single misconception (KC), we created a model that maps each mini-game question to its corresponding KC. The second model, `ProblemType`, instead maps each mini-game question to its problem type (one of `NumberLine`, `Addition`, `Bucket`, `Sorting` and `Sequence`). Table 1 shows the fit statistics results of these two models.

Table 1: Fit statistics results of the two baseline models. RMSE indicates 10-fold cross-validation root mean squared error, stratified by item. Values that indicate best fit are in bold.

Model (# of KCs)	AIC	BIC	RMSE
`DecimalMisc` (4)	30,699.27	34,379.97	0.3292
`ProblemType` (5)	**29,504.09**	**33,202.12**	**0.3231**

As can be seen, `ProblemType` outperforms `DecimalMisc` in all three metrics - AIC, BIC and RMSE. In other words, the actual problem types capture students' learning better than the underlying misconceptions. In subsequent analyses, we therefore focused on improving the `ProblemType` model. The first step is identifying potential improvements in the learning curve of each KC. In general, a good learning curve is smooth and decreasing [51]. Smoothness indicates that no step is much harder or easier than expected, and a decreasing curve shows that students were learning well and therefore made fewer errors at later opportunities [36].

From Figure 3, we observed that the learning curves of `NumberLine` and `Bucket` are reasonably good. The learning curve of `Addition` stays at roughly the same low error rate throughout (< 10%) , but there are sudden peaks, suggesting that some problems were harder than others and thus should be represented by a separate KC. The learning curve of `Sequence` decreases but not smoothly; the zigzag pattern indicates that students were alternating between easy and hard problems. Again, having separate KCs for the lows and highs of the curve would likely yield a better fit. The learning curve of `Sorting` is neither decreasing nor smooth; therefore, this KC needs to be further decomposed.

3.2 Improved KC Models

3.2.1 KC decomposition

To find possible decompositions, we followed the human-machine discovery method outlined in [51] and consulted prior literature on students' learning of decimal numbers. Below we present our analysis of each problem type.

NumberLine. As its learning curve is already good, we turned to related work on the game *Battleship Numberline* [29], where students have to place given fraction numbers on a number line. The authors found that, on a number line that runs from 0 to 1, students have better understanding when adjusting from 0 or 1 (e.g., 1/10 or 9/10) than from 1/2 (e.g., 3/7). Since decimal numbers can be translated to fractions and vice versa, we (tentatively) experimented with applying the findings of [29] to our model. In particular, we decomposed the `NumberLine` KC into `NumberLineMid` (the number to locate lies between 0.25-0.75) and `NumberLineEnd` (the number to locate lies between 0-0.25 or 0.75-1).

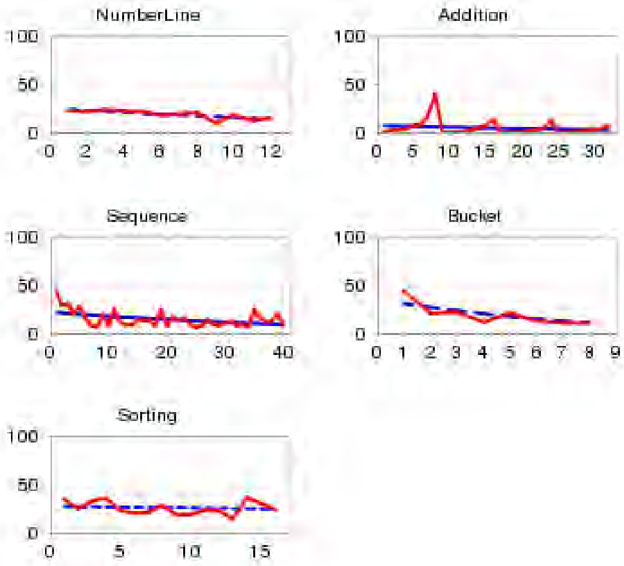


Figure 3: Learning curves of the KCs in `ProblemType`. The x-axis denotes opportunity number for each KC and y-axis denotes error rate (%). The red line plots all of the actual students' error rate at each opportunity, while the blue line is the curve fit by AFM.

Addition. There are eight items in an `Addition` game: four text boxes for carry digits - *carryTens, carryOnes, carryTenths, carryHundredths* - and four text boxes for the result - *ansTens, ansOnes, ansTenths, ansHundredths* (see Figure 2b for an example). Previously, all of these items had the same KC label of `Addition`, but we expected that some digits would be harder to compute than others. For instance, the *carryHundredths* digit is always 0, because our problems only involve numbers with two decimal places. On the other hand, because the focus of `Addition` problems is to test that students can carry from the decimal portion to the whole number portion (i.e., probing for the `Pegz` misconception), the *carryOnes* digit is always expected to be 1. It was indeed the case that *carryOnes*, along with *ansOnes*, accounts for a large portion of the peaks in `Addition`'s learning curve (Figure 3). The most common error in these peaks, however, comes from *carryTens* and *ansTens* in the mini-game *Thirsty Vampire 1*. For the majority of students in our sample (87.5%), *Thirsty Vampire 1* was the first `Addition` problem they encountered, and its question (7.50 + 3.90) was also the only one with a carry in the tens place; in other words, it was both the first and hardest question. For this reason, we decided to decompose the `Addition` KC into:

- `Addition_Tens_NonZero` applies to the *carryTens* and *ansTens* item in *Thirsty Vampire 1.*
- `Addition_Ones` applies to *carryOnes* and *ansOnes* in all `Addition` mini-games.
- Other items (e.g., *carryTenths, carryHundreds ansTenths*) retain the KC label `Addition`.

Sequence. In a `Sequence` mini-game, students have to enter the last two numbers in an increasing arithmetic sequence, based on the pattern of the first three given numbers (e.g., Figure 2a). In the way the questions were designed, the first number to fill in always requires an addition with carry, whereas the second does not involve a carry. We therefore hypothesized that the first number is more difficult than the second, which was confirmed by inspection of the learning curve: the alternate up and down patterns depict students' error rates as they filled in the first and second number in each sequence. We further distinguished between numbers with two decimal digits and those with one, as the former should be more difficult to work with. In summary, we decomposed the `Sequence` KC into four KCs: `Sequence_First_OneDigit` (first number, with one decimal digit), `Sequence_First_TwoDigits` (first number, with two decimal digits), `Sequence_Second_OneDigit` (second number, with one decimal digit), `Sequence_Second_TwoDigits` (second number, with two decimal digits).

Bucket. As the learning curve of `Bucket` is already good, we did not further decompose this KC.

Sorting. The learning curve of `Sorting` remains flat at around a 25% error rate. Since there are no outstanding blips or peaks in this curve, we instead used DataShop's Performance Profiler tool to plot the predicted and actual error rates of each mini-game problem (Figure 4). We identified five mini-game problems in which the actual error rate was larger than predicted by at least 5%; in other words, these problems were harder than expected. Therefore, we labeled five of them - *Rocket Science 1, Rocket Science 2, Jungle Zipline 2, Balloon Pop 2* and *Whac A Gopher 1* - by a separate KC called `SortingHard`, while other problems remained in `Sorting`. We will characterize the mathematical features of these `SortingHard` problems in Section 4.2.

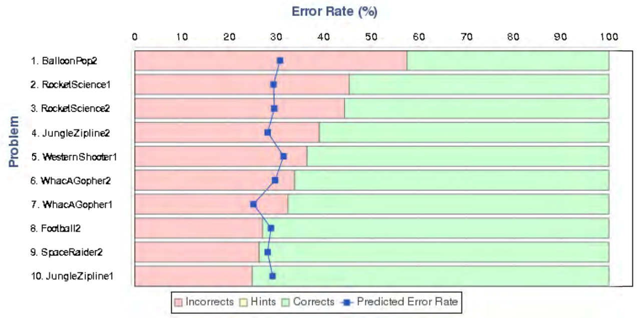


Figure 4: Visualization of the `Sorting` KC's goodness of fit with respect to ten `Sorting` mini-games with the highest error rates. The bars (shaded from left) show the actual error rates and the blue line shows predicted error rates.

3.2.2 *New model result & comparison*

Table 2 shows the fit scores of the original `ProblemType` model, the models resulting from individual KC decompositions, and the final model combining all decompositions, called `Combined`. Apart from `ProblemType` and `Combined`, the name of each other model indicates which original problem type KC is decomposed. For instance, the `Sorting` model has six KCs - `SortingHard, Sorting, NumberLine, Bucket, Addition, Sequence` - where the last four are identical to those in `ProblemType`. We can therefore see that decomposing the original `Sorting` KC alone results in a decrease of AIC by 231.91 and BIC by 214.59.

Table 2: Fit statistics results of the original and new models, sorted by AIC in descending order. Values that indicate best fit are in bold.

Model (# of KCs)	AIC	BIC	RMSE
`ProblemType` (5)	29,504.09	33,202.12	0.3231
`NumberLine` (6)	29,492.48	33,207.83	0.3233
`Sorting` (6)	29,272.18	32,987.53	0.3215
`Sequence` (8)	29,159.27	32,909.25	0.3234
`Addition` (7)	29,025.77	32,758.43	0.3235
`Combined` (12)	**28,436.07**	**32,255.34**	**0.3196**

Figure 5 shows the resulting learning curves of the above decompositions. We observed three KCs with issues: (1) `Sequence_First_TwoDigits` is a flat curve which indicates no learning, (2) `SortingHard` remains at high error rates, and (3) `Addition_Tens_NonZero` has too little data (because it only applies to *Thirsty Vampire 1*). Three other KCs - `Addition`, `Addition_Ones`, `Sequence_Second_Digits` - have low and flat curves, suggesting that students already mastered them early on and did not need as much practice (i.e., they were over-practicing with these KCs). The remaining KCs have smooth and decreasing curves. Most notably, we were able to fix the zigzag pattern in the original `Sequence` curve, reduce the peaks in the `Addition` curve, and capture the `Sorting` problems that do reflect students' learning.

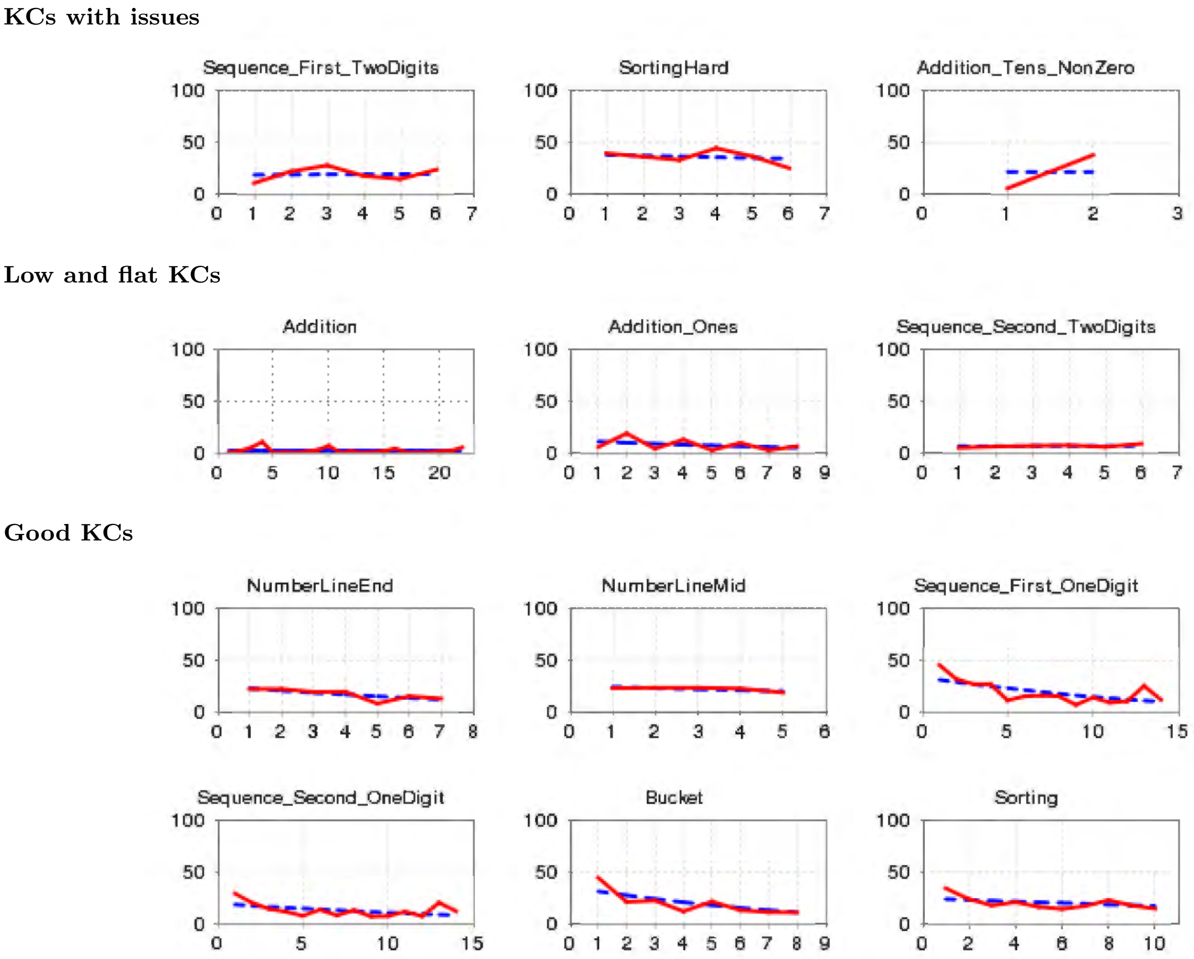
Other than `NumberLine`, all of the new models resulted in better AIC and BIC scores. The `Combined` model, which incorporates all decompositions, is the best fit; when compared to `ProblemType`, its AIC score is lower by 1068.02 and its BIC is lower by 946.78. Using DataShop's Performance Profiler tool, we were also able to visualize the differences between these models in Figure 6. Here we see that for each of the new KCs, the `Combined` model's prediction, represented by the blue line (square points), is closer to the actual error rate than the `ProblemType` model's prediction, represented by the green line (round points). Hence, the combination of our KC decompositions resulted in a better fit visually.

4. DISCUSSION

4.1 Comparison of Baseline Models

We found that the `ProblemType` model, which maps mini-game questions to problem types, is a better fit for student learning than the `DecimalMisc` model, which maps mini-game questions to underlying misconceptions. Here we outline two possible interpretations.

First, while each question was designed to test one misconception, students may demonstrate other misconceptions in their answers. For example, the mini-game *Jungle Zipline 1*, labeled as `Segz` (shorter decimals are larger), asks students to sort the decimals 1.333, 1.33, 1.3003, 1.3 from smallest to largest. An answer of 1.3003, 1.333, 1.33, 1.3 would match


KCs with issues

Low and flat KCs

Good KCs

Figure 5: Learning curves of the KCs in `Combined`. The x-axis denotes opportunity number and y-axis error rate (%). The red line plots the actual students' error rate at each opportunity, while the blue line is the curve fit by AFM.

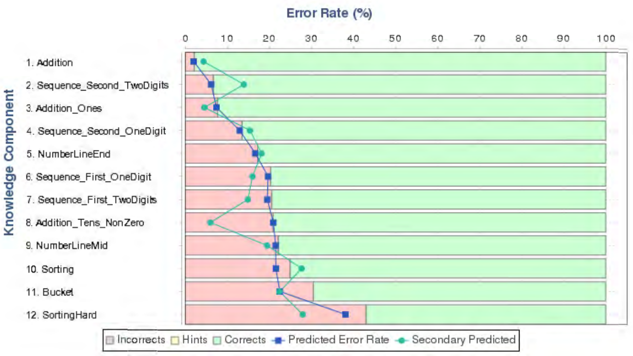


Figure 6: Visualization of the `Combined` and `ProblemType` models' goodness of fit with respect to the new KCs. The bars (shaded from left) show the actual error rates. The blue and green line show predicted error rates of `Combined` and `ProblemType` respectively.

the `Segz` misconception, but we observed that 25% of the incorrect answers were 1.3, 1.33, 1.333, 1.3003, which instead corresponds to `Megz` (longer decimals are larger). As another example, the mini-game *Capture Ghost 1*, labeled as `Megz`, asks students to decide if each of the following numbers - 0.5, 0.341, 0.213, 0.7, 0.123 - is smaller or larger than 0.51. 14% of the incorrect answers stated that 0.5 > 0.51 and also 0.341 > 0.51, which demonstrates both `Segz` and `Megz`, respectively. In general, in a problem solving environment like *Decimal Point*, measuring students' misconceptions should be based on their actual answers, not the questions alone. Therefore, a KC model that maps each question to its hypothesized misconception may not capture the students' full range of learning difficulties. Two alternative approaches used by other research for tracking decimal misconceptions are: (1) measuring them at a larger grain size, such as *whole number*, *role of zero* and *fraction* [14], and (2) using erroneous examples instead of problem solving questions [21]. In the context of KC modeling, we could apply our process to an existing dataset of student learning of decimal numbers from erroneous examples, such as the dataset from [33].

From a cognitive perspective, [44] pointed out that "different kinds of knowledge and competencies only show up intertwined in behavior, making it hard to measure them validly and independently of each other." The authors conducted a series of studies to test students' conceptual knowledge of decimal numbers and procedural knowledge of locating them on a number line. Each study employed four common hypothetical measures of each kind of knowledge, but revealed substantial problems with the measures' validity, suggesting that it is difficult to reliably separate tests of conceptual knowledge and procedural knowledge. In our context, the decimal misconceptions reflect conceptual knowledge while the problem types require a combination of both conceptual and procedural knowledge. Therefore, differentiating problems by their types creates clearer KC distinctions than by their associated misconceptions, because the former matches more closely with students' actual performance.

4.2 Interpretation of the New KCs

Here we discuss the insights from our earlier KC decomposition results, using a combination of learning curve analyses and domain-specific interpretations. While the example questions we cite are specific to those in *Decimal Point*, the findings about student learning are applicable to any other educational technology system in decimal numbers.

NumberLine. Unlike [29], we did not observe that students have more difficulty with numbers close to 0.5 than with numbers close to 0 or 1. Decomposing `NumberLine` into `NumberLineEnd` and `NumberLineMid` results in increases in BIC and RMSE, which are indicative of overfit. Furthermore, the original learning curve of `NumberLine` is already smooth and decreasing (Figure 3), so it is unlikely that any decomposition would yield significant improvements. More generally, this result suggests that students could learn to estimate the magnitude of a given decimal number between 0 and 1 reasonably well, even though they may have difficulty with the equivalent fraction form in the way [29] reported. To explain this difference, we should note that students tend not to perceive decimals and fractions as being equivalent [47], hence difficulties with fractions may not translate to difficulties with decimal numbers. As [12] pointed out, a fraction `a/b` represents both the relation between `a` and `b` and the magnitude of the division of `a` by `b`, whereas a decimal number, without the relational structure, more directly expresses a one-dimensional magnitude. Therefore, students often have higher accuracy in estimating decimal numbers than fractions on a number line [53]. The findings from our analysis and [29] further support this distinction.

Addition and Sequence. These problem types both involve computing the sum of two decimal numbers, and as our decompositions showed, the difficulty factor lies in carrying digits to the next highest place value. In the case of `Addition`, the first question, which also happens to be the most challenging, is to add 7.50 and 3.90, which requires two carries, one to the ones place and one to the tens place. The error rate is therefore highest for this question (the first peak in Figure 3), but decreases at later (easier) opportunities. The original learning curve of `Sequence` problems has a zigzag pattern due to the students alternating between additions with and without carry. Distinguishing between these two types of operations, and also on the number of decimal digits, did result in a better model fit. We also note that the error rates in `Sequence` problems are generally higher than in `Addition` problems. A possible interpretation is that, while the underlying addition operations are similar, the `Sequence` interface does not lay out the carry and result digits in detail as the `Addition` interface does (Figure 2). As pointed out by [25], for adding and subtracting decimals of different lengths, incorrect alignment of decimal operands is the most frequent source of error. Since `Addition` problems already supported this alignment via the interface, students were less likely to make mistakes in them.

Bucket and Sorting. These problem types both involve performing comparisons in a list of five decimal numbers, but in different manners. `Bucket` problems require comparing each number to a given threshold value, while `Sorting` problems require comparing the numbers among themselves. According to [40], ordering more than two decimals (`Sorting`) could reveal latent erroneous thinking which mere comparison of pairs (`Bucket`) cannot. Consistent with this finding, our results also showed that students were able to learn `Bucket` problems well but struggled with `Sorting`. Our hypothesis is that a `Sorting` problem requires two separate skills: (1) comparing individual pairs of number (in a list of five numbers, students may perform up to ten comparisons), and (2) ordering the numbers once all the comparisons have been established. The current interface only asks for the final sorted list, so it would need to be redesigned to allow for tracking student mastery of each of these two skills. Furthermore, by examining the five problems categorized as `SortingHard`, we identified unique challenges that were not present elsewhere in *Decimal Point*. First is the issue of negative number - the mini-game *Balloon Pop 2*, with an error rate close to 60% (Figure 4), asks students to sort the sequence 8.5071, -8.56, 8.5, -8.517 in descending order. Given that students may hold misconceptions about both the length and sign of decimal numbers [21], and that no other `Sorting` problems involve negative numbers, it is clear why students faced significant difficulties in this case. The second issue is another common misconception - that a 0 immediately to the right of the decimal point does not matter (e.g., 0.03 = 0.3) - which [39] referred to as *role of zero*. It could be invoked in the mini-game *Rocket Science 1*, which asks students to sort 0.14, 0.4, 0.0234, 0.323 in ascending order; in particular, 19% of the incorrect answers put 0.0234 between 0.14 and 0.323, implying the incorrect belief that 0.0234 = 0.234. Previous studies have also reported that 9th graders and even pre-service teachers demonstrated this misconception in similar sorting tasks [20, 38]. Furthermore, students may still have this misconception even after abandoning others [13].

According to [24], there are four steps to redesign a tutor based on an improved cognitive model: (1) resequencing, (2) knowledge tracing, (3) creating new tasks, and (4) changing instructional messages, hint and feedback. Based on this framework and our analyses, we derived the following lessons for designing instructional materials in our digital learning game and other tutoring systems in decimal numbers:

1. Arrange the easy `Addition` problems (without or with one carry) at the beginning. The number of these easy problems can also be reduced, as over practice is already occurring based on the number of problems students are attempting with low error rates.
2. Design more `Addition` problems with varying difficulties (those with more carries are more difficult) and

position them in increasing order of difficulty.

3. Leave the operand fields blank in Addition problems so that students can practice aligning decimal digits. Getting feedback on this alignment task could in turn help them solve Sequence problems better.
4. Provide more scaffolding in Sorting problems, by first asking students to perform pairwise comparisons of the given numbers, then having them place the numbers in order. The first task can be used to track misconceptions and the second to track the skill of ordering.
5. Design questions in other problem types besides Sorting (e.g., NumberLine, Bucket) that address the *role of zero* misconception, as it may be stronger and persist longer than other misconceptions.

4.3 Advantages of Post-hoc KC Modeling

While, in general, KC modeling methods can be applied to any domain, domain knowledge is still critical for the interpretation of the improved models and an understanding of the newly discovered KCs. We have shown that we can apply methods in a post-hoc manner to a dataset in an educational domain to both achieve a better understanding and create a better fitting KC model. Our findings also demonstrate that the type of KC modeling we used can help guide changes to the types, contents and order of problems that are used in a decimal learning game (and educational technology more generally). From a theoretical perspective, the search space for a KC model in a given domain will be somewhere between a Single KC model, where every step represents the same KC, to a Unique Step model, where every step has its own KC. If we include the option of tagging a single step with multiple KCs, the space could get infinitely larger, but in a practical sense multi-coded steps could be combined to a single KC by concatenating the KCs on a given step. Several automated processes have been applied to create KC models by searching the possible space, such as Q-Matrix search [48], but they have the limitation of creating models with unlabeled skills. The methods that we used do not face this problem because we started with a fully labeled model and worked from there. Using visual and computational analyses on the learning curves, we were able to make improvements by combining the output of fitting models with domain knowledge. The original Addition KC is an excellent example of this approach in action. While the overall curve did show a declining error rate, every four opportunities looked as if the steps were getting harder (see Figure 3). Methodologically, this was a clear opportunity for improvement and likely a feature where each successive step in a problem became harder. Sure enough, this was the case as each of four problem steps required a carry, and the hardest problem required two carries. This is one example which demonstrates that we were able to not only get a better fitting model, but also attain a deeper domain understanding.

4.4 Future Work

In our next study, we will use the best KC model from this work as a test of how well it performs with a new population of students. There is also potential in connecting our work with earlier studies of student agency in digital learning games. In particular, [37] and [19] reported that even though students in the high-agency condition could choose to play any mini-game in any order, they did not learn more than those in the low-agency condition, who played a fixed number of mini-games in a default order. [19] speculated that the former might be focused on selecting mini-games based on their visual themes (e.g., *Haunted House*, *Wild West* - see Figure 1) rather than learning content. To address this issue, we could employ an open learner model [4] that displays the estimated mastery level of each decimal skill to the students, where the skills are the KCs in our best model. In this scenario, we expect that students who exercise agency would be able to make informed selections of mini-games based on an awareness of their learning progress.

At the same time, digital learning games are intended to engage students and promote learning. Therefore, we want to explore the interactions between enjoyment and learning, particularly in how best to balance them. Just as learning can be modeled by knowledge components, can enjoyment also be modeled by "fun components," and how would they be identified? We believe our digital learning game is an excellent platform for this exploration, because each mini-game has a separate learning factor (the decimal question) and enjoyment factor (the visual theme and game mechanics). It is also possible to track students' enjoyment either through in-game surveys or automated affect detectors [1]. As our next step, we will design two study conditions, one that employs a traditional open learner model and one that captures and reflects students' enjoyment, using the five problem types (worded in a more playful way, e.g., Shooting instead of Sorting, because all Sorting mini-games involve shooting objects such as spaceship) as the initial fun components. Findings from this follow-up study would then allow us to refine our enjoyment model and provide insights into whether a learning-driven or enjoyment-driven game design yields better outcomes.

In the direction of KC modeling, as mentioned in [19] and [52], it is possible that the the game contains more learning materials than required for mastery, or that some students may have exhibited greater learning efficiency than others. With the KC model identified in this work, we can then apply Bayesian Knowledge Tracing [11] to assess students' mastery of each KC and verify the presence of learning efficiency or over-practice. Another area we plan to study is whether individual differences among the students in their gameplay and learning could lead to further improvement in predicting skill mastery based on the best-fit KC model, similar to previous research done in an intelligent tutor for genetics learning [15]. These individual differences could be accounted for by other features in the game outside of the identified cognitive-defined KCs [16].

5. CONCLUSION

Previous work has been done on refining KC models for educational systems in the manner we have shown here [51], although our research focused on the application of the refinement techniques to a digital learning game. We found that modeling KCs by problem types yields a better fit than modeling by the underlying misconceptions that were being tested. Furthermore, the refined KC model also showed us how to improve the original learning materials, in particular by focusing on the more challenging and persistent misconceptions, such as those involving multiple carries, role of zero and negative numbers. More generally, we demonstrated how learning curve analysis can be employed to perform post-hoc KC modeling in a tutoring system with various types of task. In turn, our work opens up further opportunities to explore the interaction of student models with learning, enjoyment and agency, which would ultimately contribute to the design of a learning game that can adaptively balance these aspects.

6. ACKNOWLEDGEMENTS

This work was supported by NSF Award #DRL-1238619. The opinions expressed are those of the authors and do not represent the views of NSF. Special thanks to J. Elizabeth Richey and Erik Harpstead for offering valuable feedback. Thanks to Scott Herbst, Craig Ganoe, Darlan Santana Farias, Rick Henkel, Patrick B. McLaren, Grace Kihumba, Kim Lister, Kevin Dhou, John Choi, and Jimit Bhalani, for contributions to the development of the Decimal Point game.

7. REFERENCES

[1] R. Baker, S. Gowda, M. Wixon, J. Kalka, A. Wagner, A. Salvi, V. Aleven, G. Kusbit, J. Ocumpaugh, and L. Rossi. Sensor-free automated detection of affect in a cognitive tutor for algebra. In *Educational Data Mining 2012*, 2012.

[2] R. Baker, M. J. Habgood, S. E. Ainsworth, and A. T. Corbett. Modeling the acquisition of fluent skill in educational action games. In *International Conference on User Modeling*, pages 17–26. Springer, 2007.

[3] J. Beck and X. Xiong. Limits to accuracy: How well can we do at student modeling? In *Educational Data Mining 2013*. Citeseer, 2013.

[4] S. Bull and T. Nghiem. Helping learners to understand themselves with a learner model open to students, peers and instructors. In *Proceedings of workshop on individual and group modelling methods that help learners understand themselves, International Conference on Intelligent Tutoring Systems*, volume 2002, pages 5–13. Citeseer, 2002.

[5] H. Cen, K. Koedinger, and B. Junker. Learning factors analysis–a general method for cognitive model evaluation and improvement. In *International Conference on Intelligent Tutoring Systems*, pages 164–175. Springer, 2006.

[6] H. Cen, K. R. Koedinger, and B. Junker. Is over practice necessary?-improving learning efficiency with the cognitive tutor through educational data mining. *Frontiers in artificial intelligence and applications*, 158:511, 2007.

[7] M. T. Chi, N. De Leeuw, M.-H. Chiu, and C. LaVancher. Eliciting self-explanations improves understanding. *Cognitive science*, 18(3):439–477, 1994.

[8] C. Conati, A. Gertner, and K. Vanlehn. Using bayesian networks to manage uncertainty in student modeling. *User modeling and user-adapted interaction*, 12(4):371–417, 2002.

[9] C. Conati and M. Manske. Evaluating adaptive feedback in an educational computer game. In *International workshop on intelligent virtual agents*, pages 146–158. Springer, 2009.

[10] C. Conati and X. Zhao. Building and evaluating an intelligent pedagogical agent to improve the effectiveness of an educational game. In *Proceedings of the 9th international conference on Intelligent user interfaces*, pages 6–13. ACM, 2004.

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

[12] M. DeWolf, M. Bassok, and K. J. Holyoak. From rational numbers to algebra: Separable contributions of decimal magnitude and relational understanding of fractions. *Journal of experimental child psychology*, 133:72–84, 2015.

[13] K. Durkin and B. Rittle-Johnson. The effectiveness of using incorrect examples to support learning about decimal magnitude. *Learning and Instruction*, 22(3):206–214, 2012.

[14] K. Durkin and B. Rittle-Johnson. Diagnosing misconceptions: Revealing changing decimal fraction knowledge. *Learning and Instruction*, 37:21–29, 2015.

[15] M. Eagle, A. Corbett, J. Stamper, B. M. McLaren, R. Baker, A. Wagner, B. MacLaren, and A. Mitchell. Exploring learner model differences between students. In *International Conference on Artificial Intelligence in Education*, pages 494–497. Springer, 2017.

[16] M. Eagle, A. Corbett, J. Stamper, B. M. McLaren, A. Wagner, B. MacLaren, and A. Mitchell. Estimating individual differences for student modeling in intelligent tutors from reading and pretest data. In *International Conference on Intelligent Tutoring Systems*, pages 133–143. Springer, 2016.

[17] J. P. González-Brenes and J. Mostow. Dynamic cognitive tracing: Towards unified discovery of student and cognitive models. *International Educational Data Mining Society*, 2012.

[18] E. Harpstead and V. Aleven. Using empirical learning curve analysis to inform design in an educational game. In *Proceedings of the 2015 Annual Symposium on Computer-Human Interaction in Play*, pages 197–207. ACM, 2015.

[19] E. Harpstead, J. E. Richey, H. Nguyen, and B. M. McLaren. Exploring the subtleties of agency and indirect control in digital learning games. In *International Learning Analytics & Knowledge Conference*. Springer, 2019.

[20] J. Hiebert and D. Wearne. Procedures over concepts: The acquisition of decimal number knowledge. *Conceptual and procedural knowledge: The case of mathematics*, pages 199–223, 1986.

[21] S. Isotani, D. Adams, R. E. Mayer, K. Durkin, B. Rittle-Johnson, and B. M. McLaren. Can erroneous examples help middle-school students learn decimals? In *European Conference on Technology Enhanced Learning*, pages 181–195. Springer, 2011.

[22] K. R. Koedinger, A. T. Corbett, and C. Perfetti. The knowledge-learning-instruction framework: Bridging the science-practice chasm to enhance robust student learning. *Cognitive science*, 36(5):757–798, 2012.

[23] K. R. Koedinger, E. A. McLaughlin, and J. Stamper. Automated student model improvement. *International Educational Data Mining Society*, 2012.

[24] K. R. Koedinger, J. C. Stamper, E. A. McLaughlin, and T. Nixon. Using data-driven discovery of better student models to improve student learning. In *International Conference on Artificial Intelligence in*

Education, pages 421–430. Springer, 2013.

[25] M. Y. Lai and S. Murray. What do error patterns tell us about hong kong chinese and australian students: understanding of decimal numbers? 2014.

[26] R. V. Lindsey, M. Khajah, and M. C. Mozer. Automatic discovery of cognitive skills to improve the prediction of student learning. In *Advances in neural information processing systems*, pages 1386–1394, 2014.

[27] R. Liu and K. R. Koedinger. Closing the loop: Automated data-driven cognitive model discoveries lead to improved instruction and learning gains. *Journal of Educational Data Mining*, 9(1):25–41, 2017.

[28] R. Liu, E. A. McLaughlin, and K. R. Koedinger. Interpreting model discovery and testing generalization to a new dataset. In *Educational Data Mining 2014*. Citeseer, 2014.

[29] D. Lomas, D. Ching, E. Stampfer, M. Sandoval, and K. Koedinger. " battleship numberline": A digital game for improving estimation accuracy on fraction number lines. *Society for Research on Educational Effectiveness*, 2011.

[30] M. Manske and C. Conati. Modelling learning in an educational game. In *AIED*, pages 411–418, 2005.

[31] J. Martin and K. VanLehn. Student assessment using bayesian nets. *International Journal of Human-Computer Studies*, 42(6):575–591, 1995.

[32] R. E. Mayer. *Computer games for learning: An evidence-based approach*. MIT Press, 2014.

[33] B. M. McLaren, D. M. Adams, and R. E. Mayer. Delayed learning effects with erroneous examples: a study of learning decimals with a web-based tutor. *International Journal of Artificial Intelligence in Education*, 25(4):520–542, 2015.

[34] B. M. McLaren, D. M. Adams, R. E. Mayer, and J. Forlizzi. A computer-based game that promotes mathematics learning more than a conventional approach. *International Journal of Game-Based Learning (IJGBL)*, 7(1):36–56, 2017.

[35] K. Muldner and C. Conati. Evaluating a decision-theoretic approach to tailored example selection. In *IJCAI*, pages 483–488, 2007.

[36] A. Newell and P. S. Rosenbloom. Mechanisms of skill acquisition and the law of practice. *Cognitive skills and their acquisition*, 1(1981):1–55, 1981.

[37] H. Nguyen, E. Harpstead, Y. Wang, and B. M. McLaren. Student agency and game-based learning: A study comparing low and high agency. In *International Conference on Artificial Intelligence in Education*, pages 338–351. Springer, 2018.

[38] I. J. Putt. Preservice teachers ordering of decimal numbers: When more is smaller and less is larger!. *Focus on learning problems in mathematics*, 17(3):1–15, 1995.

[39] L. B. Resnick, P. Nesher, F. Leonard, M. Magone, S. Omanson, and I. Peled. Conceptual bases of arithmetic errors: The case of decimal fractions. *Journal for research in mathematics education*, pages 8–27, 1989.

[40] A. Roche and D. M. Clarke. When does successful comparison of decimals reflect conceptual understanding. *Mathematics education for the third millennium: Towards 2010*, pages 486–493, 2004.

[41] D. Rohrer, R. F. Dedrick, and K. Burgess. The benefit of interleaved mathematics practice is not limited to superficially similar kinds of problems. *Psychonomic bulletin & review*, 21(5):1323–1330, 2014.

[42] J. P. Rowe and J. C. Lester. Modeling user knowledge with dynamic bayesian networks in interactive narrative environments. In *Sixth AI and Interactive Digital Entertainment Conference*, 2010.

[43] R. J. Salden, V. A. Aleven, A. Renkl, and R. Schwonke. Worked examples and tutored problem solving: redundant or synergistic forms of support? *Topics in Cognitive Science*, 1(1):203–213, 2009.

[44] M. Schneider and E. Stern. The developmental relations between conceptual and procedural knowledge: A multimethod approach. *Developmental psychology*, 46(1):178, 2010.

[45] J. M. Schraagen, S. F. Chipman, and V. L. Shalin. *Cognitive task analysis*. Psychology Press, 2000.

[46] V. J. Shute, L. Wang, S. Greiff, W. Zhao, and G. Moore. Measuring problem solving skills via stealth assessment in an engaging video game. *Computers in Human Behavior*, 63:106–117, 2016.

[47] K. Stacey, S. Helme, and V. Steinle. Confusions between decimals, fractions and negative numbers: A consequence of the mirror as a conceptual metaphor in three different ways. In *PME CONFERENCE*, volume 4, pages 4–217, 2001.

[48] J. Stamper, T. Barnes, and M. Croy. Extracting student models for intelligent tutoring systems. In *Proceedings of the National Conference on Artificial Intelligence*, volume 22, page 1900. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2007.

[49] J. Stamper, K. Koedinger, R. Baker, A. Skogsholm, B. Leber, J. Rankin, and S. Demi. Pslc datashop: A data analysis service for the learning science community. In *International Conference on Intelligent Tutoring Systems*, pages 455–455. Springer, 2010.

[50] J. Stamper, K. Koedinger, and E. Mclaughlin. A comparison of model selection metrics in datashop. In *Educational Data Mining 2013*, 2013.

[51] J. Stamper and K. R. Koedinger. Human-machine student model discovery and improvement using datashop. In *International Conference on AI in Education*, pages 353–360. Springer, 2011.

[52] Y. Wang, H. Nguyen, E. Harpstead, J. Stamper, and B. M. McLaren. How does order of gameplay impact learning and enjoyment in a digital learning game? In *International Conference on Artificial Intelligence in Education*. Springer, 2019.

[53] Y. Wang and R. S. Siegler. Representations of and translation between common fractions and decimal fractions. *Chinese Science Bulletin*, 58(36):4630–4640, 2013.

[54] M. Wilson and P. De Boeck. Descriptive and explanatory item response models. In *Explanatory item response models*, pages 43–74. Springer, 2004.